Hindawi
Journal of Sensors
Volume 2022, Article ID 8668149, 10 pages
https://doi.org/10.1155/2022/8668149

Research Article

Surface Defect Detection with Modified Real-Time Detector YOLOv3

Zhihui Wang,[1] **Houying Zhu**,[2] **Xianqing Jia**,[1] **Yongtang Bao**,[1] **and Changmiao Wang**[3]

[1]*College of Computer Science and Engineering, Shandong University of Science and Technology, Qingdao 266510, China*
[2]*Department of Mathematics and Statistics, Faculty of Science and Engineering, Macquarie University, NSW 2109, Australia*
[3]*Shenzhen Research Institute of Big Data, Shenzhen 518172, China*

Correspondence should be addressed to Changmiao Wang; wangcm@sribd.cn

Received 28 September 2021; Revised 23 March 2022; Accepted 6 April 2022; Published 30 June 2022

Academic Editor: Sangsoon Lim

In this paper, a modified YOLOv3 net has been proposed for surface defect detection. Different from other pixel-level segmenting methods, YOLOv3 locates the regions of surface defects with bounding rectangles. Compared with conventional detectors, the operating efficiency of YOLOv3 is rather high without generating region proposals by sliding boxes. Although pixel-level details of defects are omitted in the process, the primary information of the location of detects and class labels are extracted by YOLOv3 with high accuracy. This information is sufficient for surface defect inspection, and computational efficiency has been improved, simultaneously. To further light the structure of YOLOv3, loss function optimization and pruning strategy have been adopted in the original YOLOv3. The pruning ratio is determined by the tradeoff between detecting accuracy and computational efficiency. In our experiments, we compared the performance of modified YOLOv3 with several state-of-the-art methods, and modified YOLOv3 achieves the best performance on six types of surface defects in DAGM 2007 dataset.

1. Introduction

Nowadays, artificial intelligence [1–3] plays an important role in various kinds of social applications. In the field of industrial automation, effective surface defection is crucial in the quality control of the industrial environment [4]. The traditional manual detection method is time-consuming, and it does not apply to large scale products. Another primary source of difficulty in manual detection is that many of the factors of variation influence the detection accuracy. Deep learning, which has been gaining tremendous amounts of traction in recent years, especially convolutional neural networks (CNN), has achieved great success in many computer vision tasks such as image detection and classification [5]. Following recent advances in deep learning, CNN is now commonly employed to target industrial inspection tasks.

Deep learning algorithms make use of the appropriate classifiers to solve the detection and classification of defects. Deep convolutional neural networks proposed in [6] is one of the breakthrough methods for image classification for high-complexity datasets. Several deep learning methods are used on the topic of surface defect inspection. Fully convolutional network (FCN) combines a segmentation stage and a detection stage, which are operated with two separate fully convolutional networks [7, 8]. ViDi is the first deep learning-based image analysis software designed specifically for factory automation [9] and also referred to in [10]. Qiu et al. has proposed a pixel-wise surface defect segmentation method, which contains a segmentation stage, a detection stage, and a matting stage [11]. Deep recurrent neural network (DRNN) consists of four modules: deep regression-based detection model, pixel-level false positive reduction, connected component analysis, and deep network for defect

type classification [10]. It can also be simplified to be a detector without the procedure of classification. These methods have been tested on the datasets of surface defect inspection and play certain effects on them.

A central family of automatic object detection algorithms is a region-based approach. Regionlets proposed in [12] are the subregions denoted in the feature extraction region to capture the possible locations of the target object. It first constructs a largely over-complete regionlet pool, and then each object bounding box is classified by a cascaded boosting classifier. On the other hand, CNN-based methods R-CNN [13] and its modified versions Fast R-CNN [14] and Faster R-CNN [15] use regions to hypothesize object locations within the image, where bounding box around the object of interest is created to lock the position. Note that Faster R-CNN speeds up the detection using neural networks to propose bounding boxes instead of selective search as used in Fast R-CNN. Generally, CNN-based methods work reasonably well on various kinds of targets [16]. However, their operational efficiency and detecting accuracy still need to be further enhanced.

Unlike the anchor based algorithms, YOLO (You Only Look Once) is a real-time object detection system proposed by Redmon et al. in [17]. It frames object detection as a regression problem, and a single convolutional network simultaneously predicts multiple bounding boxes and class probabilities for those boxes, which makes YOLO extremely fast. To improve the location accuracy and recalling rate, Redmon and Farhadi in [18] came up with a better, faster, and stronger YOLO that can detect over 9000 different object categories, named as YOLO 9000. In this version, the fully connected layers from YOLO are removed, and anchor boxes are used to predict bounding boxes. The most advanced version of YOLO is known as YOLOv3 [19], where Darknet-53 is used to detect the objects via different scales of layers. Another appealing feature of YOLOv3 is that the end-user can simply change the model size to make the trade-off between detection accuracy and speed without retraining the model.

For the detecting tasks, objects with similar sizes and appearances are easily detected. For categories with large scale variances and changeable appearances, the detection accuracies are limited. Therefore, specific processing mechanisms need to be designed to adjust these variations. In this paper, a new item of loss function based on generalized intersection over union (GIoU loss) is designed to deal with these categories with large scale variations on training phase. Compared with original GIoU loss, the modified item pays more attention on the area difference between the predicted objects and ground truth. Furthermore, pruning categories are adopted to enhance operating efficiency. The running speed is increased largely with certain pruning percentages on network parameters.

The rest of the paper is organized as follows: We first give a brief review of the real-time object detection system YOLO and introduce the loss function optimization and pruned strategy. Then, the modified YOLOv3 net is applied to a popular benchmark dataset for multiclass object detection, DAGM 2007, wherein the full version of YOLOv3 and pruned version are compared with previous methods, including FCN [7, 8], ViDi [9], Qiu [11], DRNN [10] and its simplified version, and several variants of original YOLOv3. We wrap up with a general discussion and potential future works to defect detection.

2. Modification of YOLOv3

The series of YOLO networks are rather popular in the field of object detection, and these latest versions are powerful with thousands of categories. However, latest versions are good at extracting differentiation of details among different categories, and their structures are complex than typical version YOLOv3. These complex structures are easy to prone to overfitting issues without large amounts of training data. For example, the latest YOLOX network [20] is still using the backbone network of YOLOv3. For the applications of surface defect detection, less quantity of defect categories and samples are analyzed in each real application. Considering these above factors, YOLOv3 is chosen to be modified in this paper with certain training samples, considering of effectiveness and efficiency.

Different from other detecting methods, the anchor free strategy of YOLO finds targets without generating region proposals by sliding boxes. Since multiple independent logistic classifications and binary cross-entropy loss have been adopted in YOLOv3, the classification mechanism is enhanced than previous versions of YOLO networks. In the prediction, various scales of predicted boxes are used with different feature extraction structures. The information included each bounding box is one-time extracted, such as the potential location of objects and the corresponding class probabilities.

The backbone of adopted YOLOv3 in this paper is Darknet-53, and multiple residual blocks are combined to extract feature. The total number of layers of the YOLO is 53. Each convolutional/residual block contains convolutional filters with a size of 1×1 and 3×3, batch normalization, and leaky ReLU. These residual blocks guarantee the convergence ability of the network with deep structures. The pipeline of YOLO is as follows. For a given input image, it first rescales the input image to be 416×416. The detection of objects is operated on different layers assigned with three types of scales. The 13×13 layer is used to detect large objects, the 26×26 layer detecting medium objects, and the 52×52 layer is responsible for the smaller size objects, which significantly improve the performance for smaller objects detection compared with YOLO 9000.

In the modified structure of YOLOv3 in this paper, loss function optimization and network pruning strategy are adopted to adapt requirements of surface defect inspection, which is demonstrated in following subsections.

2.1. Loss Function. YOLOv3 further enhances the performance from the previous YOLO versions YOLOv1 and YOLO 9000 replacing softmax loss by Logistic loss, i.e., object confidence and class predictions in YOLOv3 are predicted through logistic regression. Although pixel-wise information of these defect regions is not segmented in

detail, the bounding rectangles are detected and classified in high accuracy. This information extracted from the YOLOv3 is quite useful in the accessorial work of defect inspection in the assembly line. However, the estimated rectangles are not adaptive for these situations with large scale variations of different targets.

In this paper, modified loss functions are proposed to enhance the performances of YOLOv3. The newly added item of loss function is optimized based on generalized intersection over union (GIoU loss) [21]. Based on GIoU loss, the area proportion between ground truth and corresponding estimated rectangle is considered in the item.

The modified GIoU loss is defined as

$$L_{area} = 1 - \frac{S_{pred}}{S_{pred} + S_{true}} \cdot GIoU, \tag{1}$$

with

$$GIoU = IoU - \frac{S_{conv} \setminus \left| S_{pred} \cup S_{true} \right|}{\left| S_{conv} \right|}, \tag{2}$$

where S_{true} is area of the target, S_{pred} is the estimated area by YOLOv3, and S_{conv} is the smallest convex object of them, and

$$IoU = \frac{\left| S_{pred} \cap S_{true} \right|}{\left| S_{pred} \cup S_{true} \right|}. \tag{3}$$

Since the intersection over union focuses on overlap ratio, this factor causes the compromise phenomenon that the estimated rectangles are always smaller than large targets and larger than the small targets. Although GIoU is not sensitive to sizes of object, this phenomenon is still exists. In the surface defect inspection, most of these types of surface defects are small, and few types are quite larger than others, and their estimated rectangles are smaller than areas of the targets in most cases. The added area proportion is used to improve this issue and lead the estimated rectangle to match the size of the target more properly.

With the modified GIoU loss, the total loss function is optimized as

$$L_{total} = L_{org} + L_{area}, \tag{4}$$

where L_{org} is the original loss function used in normal YOLOv3 [19], which is focusing on the accuracies of bounding box prediction, class prediction, and predictions across scales. The added item L_{area} is used to further enhance the evaluating criterion on area ratio and intersection over union.

2.2. YOLO Pruning. To guarantee processing abilities to various types of targets, deep learning nets always contain complex structures and a large number of parameters. Therefore, computing resources should be secured to computational efficiency. Promotion to terminal devices has been limited with high hardware costs. Therefore, effective optimization schemes are required to light the structures of deep learning nets and change the model size to trade-off accuracy and speed without overhead training new models.

There are several strategies to optimize deep learning nets, such as weight quantization [22], branch pruning [23, 24], and model compression [25]. With few accuracy loss, the number of parameters can be decreased to a large degree, and computational efficiency has been enhanced.

In this paper, branch pruning strategy is adopted to optimize YOLOv3. Based on the pruning criterion in [24], original YOLOv3 are pruned. The training objective is set to be

$$L_{prun} = \sum_{(x,y)} L_{total}(f(x, W), y) + \lambda \sum_{\gamma \in \Gamma} g(\gamma), \tag{5}$$

where (x, y) is the input and target of training samples, W is trainable weight of YOLOv3, $g(\cdot) = |\cdot|$ is a sparsity-introduced L_1-norm penalty on the scaling factor γ of batch normalization, and λ is a trade-off factor between the two terms. For each channel, a scaling factor γ is set and used to multiply with the channel's output. Then, the network weights and scaling factors are trained jointly with sparsity regularization. These channels in convolutional layers with small scaling factor values are pruned. Then, the compact model is fine-tuned to achieve comparable accuracy as the original full network.

Without any special libraries/hardware for efficient inference, the number of parameters, memory requirements, and computing operations of the original YOLOv3 has been reduced, simultaneously.

3. Experiments

In the field of surface defect inspection, the surface defects are always not easy to be found under open environments. Therefore, the surface images are captured with specific light source under closed environments without external influences, which is the mainstream operating way in real applications. In the experiments, two typical open source datasets are used to verify the performance of the modified algorithm, which contain DAGM 2007 (http://resources.mpi-inf.mpg.de/conferences/dagm/2007/prizes.html) and GC10-DET (https://github.com/lvxiaoming2019/GC10-DET-Metallic-Surface-Defect-Datasets) [26]. Both of them are captured with special operations under closed environments. To see the performance of defect inspection, the proposed methods, noted as M-YOLOv3 (full/pruned), are compared with other state-of-the-art detectors, which contain the YOLOv3 [19] and its pruned version [24], FCN [8], ViDi [9], Qiu [11], DRNN, and its simplified version with no classifier [10].

3.1. Experimental Study. DAGM 2007 is one of these typical datasets on the field of surface defects with visual information. This dataset contains ten types of different surface defects, and the first six types are used for public test. For this dataset, each image represents one surface sample, which may contain one defect or not. In our experiments, we test the first six types of DAGM 2007. To guarantee a

sufficient quantity of experimental samples, all training samples with surface defects are chosen in the training phase. The testing samples with surface defects are split into two subsets: The first subset is used for validation in the training phase, and the other subset is used to test the performance of the trained algorithm. The quantities of samples in training and testing phases are 618 and 297, respectively. All these training and testing samples with defects have participated in our experiments uniformly, and the results are more convincing than fewer samples. For each image, the regions of detected rectangles with the same labels are merged as the region of surface defect.

GC10-DET is more challenging with more complex situations, which is collected in a real industry and contains ten types of different surface defects and released in 2019. On each image, the number and type of defects are not limited. These defect scales are varied largely than DAGM 2007. In our experiments, all of these ten types are tested. The data processing method are similar as DAGM 2007. The quantities of samples of GC10-DET in training and testing phases are 1583 and 689, respectively.

In our experiments, detecting accuracy is evaluated by several evaluation metrics, which contains

$$\begin{aligned} Precision &= \frac{TP}{TP + FP}, \\ Recall &= \frac{TP}{TP + FN}, \\ F1 &= \frac{Precision * Recall}{Precision + Recall}, \end{aligned} \tag{6}$$

where TP, FP, and FN are true positive, false positive, and false negative, respectively.

The IOU (intersection over the union) value is estimated by the ratio between overlap and union of detected results and marked rectangles of ground truths. Based on the situation of IOU, AR column is defined as

$$AR = 2\int_{0.5}^{1} Recall\,(\mathrm{o})\,\mathrm{do} = \frac{2}{n}\sum_{i=1}^{n} \max\left(IOU(gt_i) - 0.5, 0\right), \tag{7}$$

where n is the number of defects and gt_i is the i-th ground truths.

Further, mAP (mean average precision) is calculated by the mean value of AP (average precision), and AP is equal to the area below the precision-recall curve [27].

Different from the previous segmenting strategies, we choose typical detecting rectangles to represent the region of surface defects. This method of representation can enhance detecting efficiency, sharply. The simulation has been operated on a desktop with CPU 2.40GHz, 19.19 GB RAM, and 11.02 GB GPU with GeForce-RTX-2080-Ti. This project is mainly developed on PyTorch 3.7. In practice, the running times of the proposed M-YOLOv3 (full) on DAGM and GC10-DET are 29.84 fps and 28.30 fps, respectively. The running times of the proposed M-YOLOv3 (pruned, 15%) on DAGM and GC10-DET are 35.65 fps and 35.64 fps, respectively. Since the running speed is rather fast and estimated running time is not unstable, the running times in these experiments are averaged of ten times on all testing samples. The total parameters' amount is 235.15 MB of M-YOLOv3 (full), and 15% parameters have been pruned by M-YOLOv3 (pruned, 15%). The detecting efficiency is better than most of these algorithms in Table 1, with less computing resources.

3.2. Result Comparison. The evaluation results of defect detection accuracy on DAGM 2007 are shown in Table 1. In our simulations, the full version of the modified YOLOv3 is noted as M-YOLOv3 (full), and the pruned modified version with pruning ratio η_1% is noted as M-YOLOv3 (pruned, η_1%). The full version of the original YOLOv3 is noted as YOLOv3 (full), and the pruned original version with pruning ratio η_2% is noted as YOLOv3 (pruned, η_2%). In the proposed M-YOLOv3 (full/pruned), the added item in loss function$L_{area}$is simplified as the GIoU function, with the selected optimized weight parameter 0.75 instead of area ratio$S_{pred}/(S_{pred} + S_{true})$in Equation (1), is constructed to compare with M-YOLOv3 and noted as YOLOv3-giou.

3.2.1. Performance Comparisons on DAGM. During the comparison, the M-YOLOv3 (full) and YOLOv3 (full) obtain the best and second-best overall performances on these evaluating criteria. The values of precision, recall, and F1 are even all larger than 97% for M-YOLOv3 (full) and better than 94% for YOLOv3 (full). M-YOLOv3 (pruned, 15%) is also better than YOLOv3 (pruned, 3%) on calculation quantity with similar performance and better accuracies than YOLOv3 (pruned, 15%) with similar calculation quantity. The YOLOv3-giou (pruned, 15%) has better performance than YOLOv3 (pruned, 15%) and worse than then M-YOLOv3 (pruned, 15%). The detecting accuracy of FCN, ViDi, and Qiu is rather poor in terms of AR and MeanIOU. The performances of DRNN (full) and DRNN (no classifier) are better than these three detectors but worse than the M-YOLOv3 (full) and M-YOLOv3 (pruned).

The performances of M-YOLOv3 (pruned) are even better than DRNN (full). Although pixel-level segmentation is not adopted in M-YOLOv3, the detected rectangles are still reliable in the process of locating these defects. Therefore, the M-YOLOv3 (pruned, 15%) net is more practical both on detecting precision and computational efficiency.

More details of M-YOLOv3 (full/pruned) and YOLOv3 (full/pruned) on each class of defects in DAGM 2007 are demonstrated in Tables 2, 3, and 4. The M-YOLOv3 (full) net achieves excellent accuracy on all these classes and YOLOv3 (full) net works well on Class 1/2/4/5. Since the areas of defects on Class 3 are rather small, the boundaries of detected regions are larger than ground truths for YOLOv3 (full) net. In contrast, the defects in Class 6 are even several times over other types of defects. These classes with unregular scales are well balanced by M-YOLOv3 (full) net. For M-YOLOv3 (pruned, 15%) (Table 3), most of the values on these terms are larger than 95% with few dropped ratios. Compared with M-YOLOv3 (full) net, some values

Table 1: Result comparison of defect detection accuracy.

Method	AR	MeanIOU	Precision	Recall	F1
FCN [8]	0.6062	0.6654	—	—	—
ViDi [9]	0.5604	0.6199	—	—	—
Qiu [11]	0.6843	0.7326	—	—	—
DRNN (full) [10]	0.6902	0.8450	0.9250	0.9080	0.9150
DRNN (no classifier) [10]	0.6366	0.8172	0.8910	0.9080	0.8970
YOLOv3 (full)	0.7613	0.8806	0.9439	0.9883	0.9654
YOLOv3 (pruned, 3%)	0.7099	0.8703	0.9154	0.9595	0.9362
YOLOv3 (pruned, 15%)	0.0978	0.2821	0.3770	0.2549	0.2705
YOLOv3-giou (pruned, 15%)	0.7672	0.8819	0.9435	0.9348	0.9372
M-YOLOv3 (full)	0.7825	0.8910	0.9723	0.9867	0.9794
M-YOLOv3 (pruned, 15%)	0.7848	0.8924	0.9651	0.9398	0.9513

Note that parts of these results are cited from [10].

Table 2: Detection accuracy of M-YOLOv3 (full) and YOLOv3 (full).

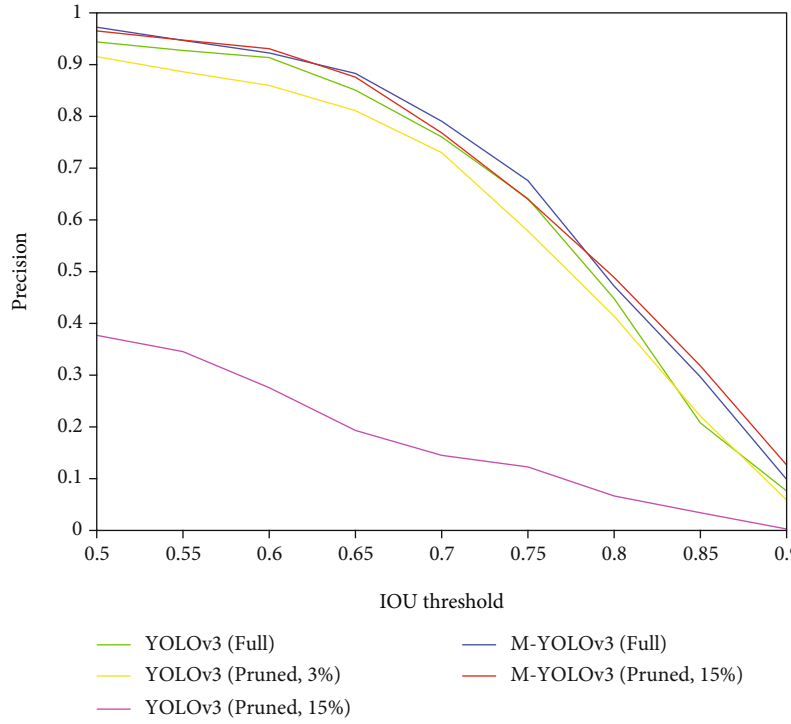
Types	M-YOLOv3 (full)				YOLOv3 (full)			
	Precision	Recall	F1	mAP	Precision	Recall	F1	mAP
Class 1	0.9574	0.9783	0.9677	0.9664	1.000	0.9787	1.000	0.9892
Class 2	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.9955
Class 3	0.9649	0.9649	0.9649	0.9598	0.8689	0.9298	0.8983	0.8996
Class 4	0.9744	1.000	0.9870	0.9987	1.000	0.9500	1.000	1.000
Class 5	0.9825	1.000	0.9912	1.000	0.9825	1.000	0.9912	1.000
Class 6	0.9545	0.9767	0.9655	0.9647	0.9348	1.000	0.9663	0.9980

Table 3: Detection accuracy of M-YOLOv3 (pruned, 15%) and YOLOv3-giou (pruned, 15%).

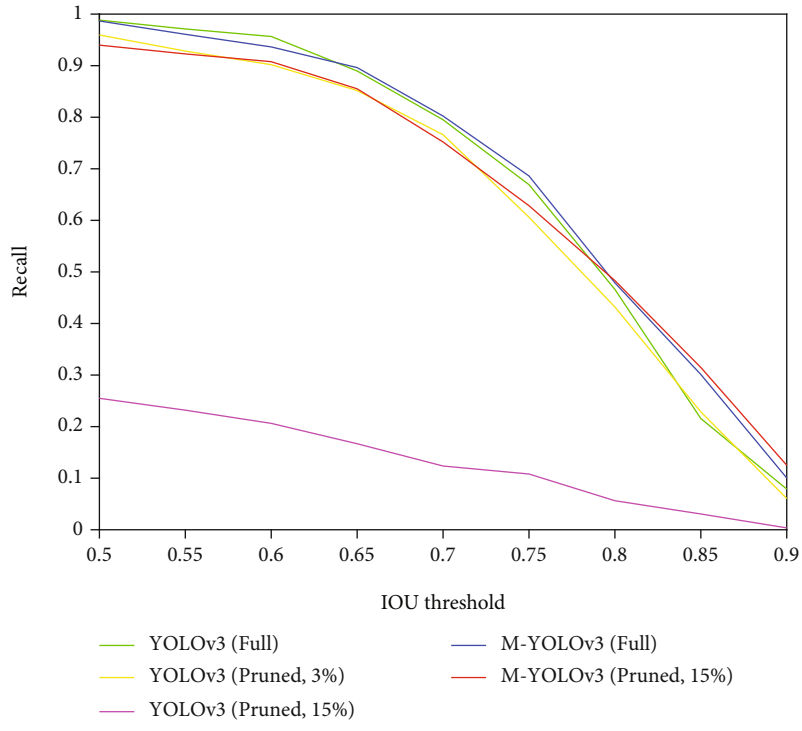
Types	M-YOLOv3 (pruned, 15%)				YOLOv3-giou (pruned, 15%)			
	Precision	Recall	F1	mAP	Precision	Recall	F1	mAP
Class 1	0.9565	0.9565	0.9565	0.9366	1.000	1.000	1.000	1.000
Class 2	0.9828	1.000	0.9913	0.9927	1.000	0.9825	0.9912	0.9082
Class 3	0.8793	0.8947	0.8870	0.8349	0.9000	0.9474	0.9231	0.9082
Class 4	1.000	0.9737	0.9867	0.9737	0.9474	0.9474	0.9474	0.9453
Class 5	1.000	1.000	1.000	1.000	0.8710	0.9643	0.9153	0.9194
Class 6	0.9722	0.8140	0.8861	0.8127	0.9429	0.7674	0.8462	0.7395

Table 4: Detection accuracy of YOLOv3 (pruned, 3%) and YOLOv3 (pruned, 15%).

Types	M-YOLOv3 (pruned, 15%)				YOLOv3-giou (pruned, 15%)			
	Precision	Recall	F1	mAP	Precision	Recall	F1	mAP
Class 1	0.9565	0.9565	0.9565	0.9366	1.000	1.000	1.000	1.000
Class 2	0.9828	1.000	0.9913	0.9927	1.000	0.9825	0.9912	0.9082
Class 3	0.8793	0.8947	0.8870	0.8349	0.9000	0.9474	0.9231	0.9082
Class 4	1.000	0.9737	0.9867	0.9737	0.9474	0.9474	0.9474	0.9453
Class 5	1.000	1.000	1.000	1.000	0.8710	0.9643	0.9153	0.9194
Class 6	0.9722	0.8140	0.8861	0.8127	0.9429	0.7674	0.8462	0.7395

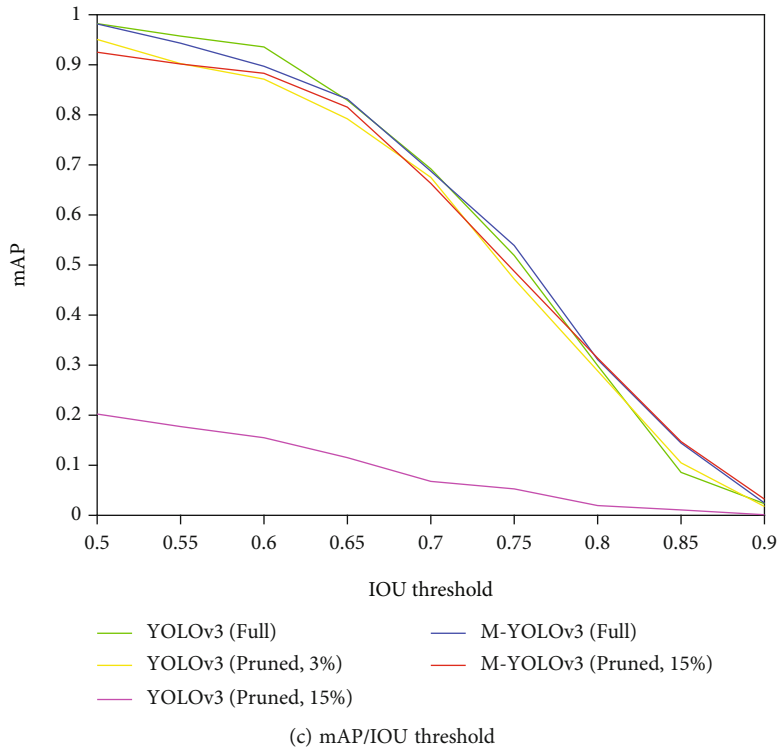


(a) Precision/IOU threshold



(b) Recall/IOU threshold

Figure 1: Continued.



(c) mAP/IOU threshold

Figure 1: Precision, recall, and mAP with respect to IOU threshold on DAGM 2007 dataset using YOLOv3.

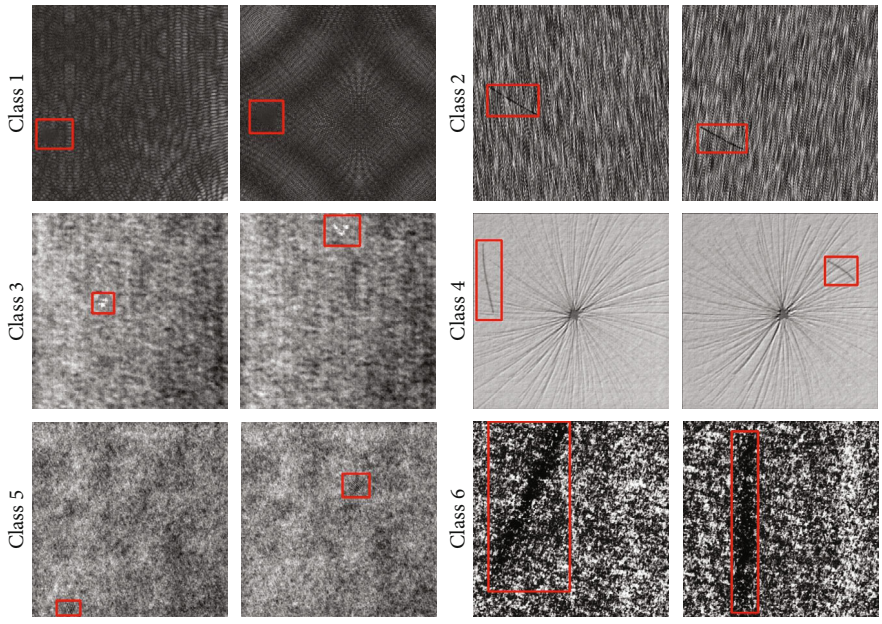


Figure 2: Examples of detected defects in DAGM 2007 dataset with M-YOLOv3 (pruned, 15%).

have even appeared with a slight increase on Precision of Class 4/5/6. Only the recall, F1, and mAP turn to worse with certain dropped ratios, which is still better than YOLOv3-giou (pruned, 15%) with same pruning ratio. The YOLOv3-giou (pruned, 15%) net has competitive performances on Class 1/2/3 and turned to be poor on other three classes. For YOLOv3 (pruned, 3%) in Table 4, precision on Class 6 is even dropped with 9.48%. For YOLOv3 (pruned, 15%), all these values on four items are dropped with large dropped ratios.

As shown in Figure 1, the differences between the performances of M-YOLOv3 (full) and M-YOLOv3 (pruned, 15%) are not significant. Most of these values in terms of precision/recall/mAP are similar. The majority of these detected

Table 5: Detection accuracy of M-YOLOv3 (full/pruned) and YOLOv3 (full).

Types	M-YOLOv3 (full)				YOLOv3 (pruned, 15%)				YOLOv3 (full)			
	Precision	Recall	F1	mAP	Precision	Recall	F1	mAP	Precision	Recall	F1	mAP
Pu	0.6148	0.9121	0.7345	0.6284	0.6484	0.9121	0.7580	0.6699	0.5321	0.9121	0.6721	0.6039
Wl	0.6036	0.8645	0.7109	0.5966	0.5575	0.8129	0.6614	0.5997	0.5000	0.7806	0.6096	0.5200
Cg	0.8214	0.9200	0.8679	0.8508	0.8023	0.9200	0.8571	0.8554	0.7473	0.9067	0.8193	0.8138
Ws	0.6303	0.7576	0.6881	0.6862	0.5726	0.7172	0.6368	0.5774	0.5833	0.7071	0.6393	0.5794
Os	0.3015	0.5698	0.3944	0.2718	0.2530	0.4884	0.3333	0.2222	0.2265	0.5174	0.3150	0.1751
Ss	0.4375	0.4941	0.4641	0.3066	0.5115	0.5255	0.5184	0.3485	0.3636	0.4588	0.4055	0.2656
In	0.1295	0.2479	0.1701	0.0891	0.1250	0.2393	0.1642	0.0506	0.0858	0.1709	0.1143	0.0242
Rp	0.1667	0.0833	0.1111	0.0152	0.1667	0.2083	0.1852	0.0361	0.5000	0.1250	0.2000	0.0903
Cr	0.3500	0.2414	0.2857	0.1636	0.0789	0.1034	0.0896	0.0182	0.3077	0.1379	0.1905	0.1141
Wf	0.7273	0.5854	0.6486	0.5033	0.6897	0.4878	0.5714	0.4112	0.5862	0.4146	0.4857	0.3224
Overall	0.4783	0.5676	0.5075	0.4111	0.4406	0.5415	0.4775	0.3789	0.4432	0.5131	0.4451	0.3509

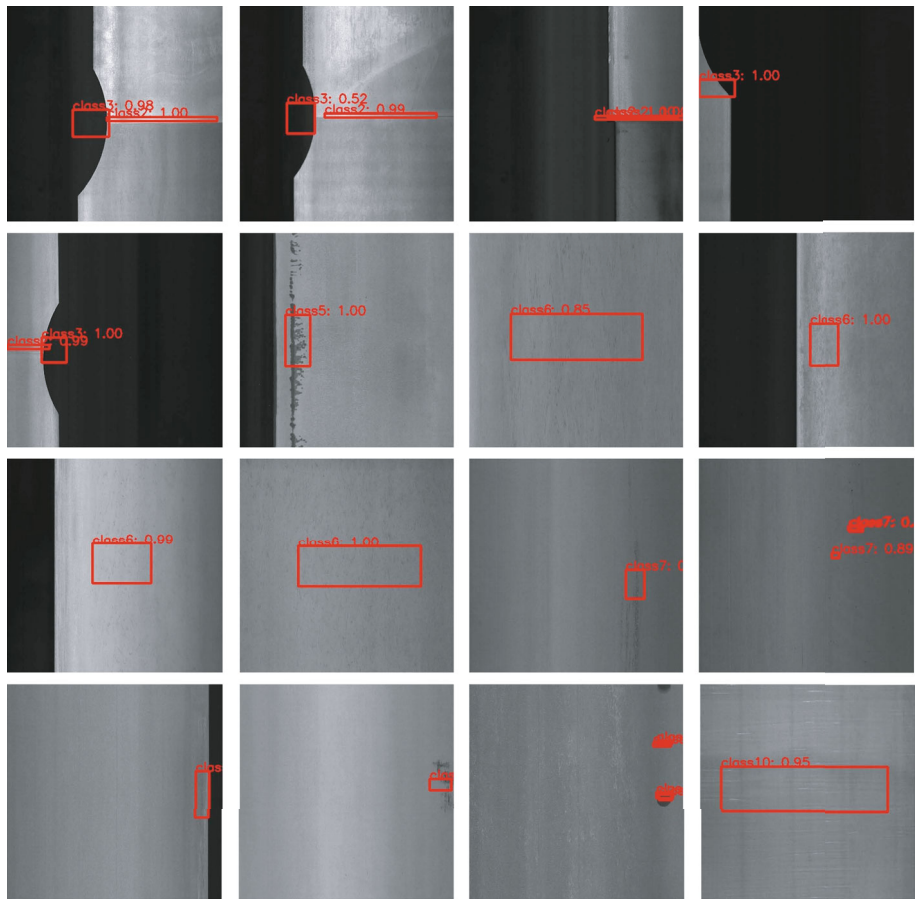


Figure 3: Examples of detected defects in GC10-DET dataset with M-YOLOv3 (pruned, 15%). The type ID from class 1 to class 10: Pu, Wl, Cg, Ws, Os, Ss, In, Rp, Cr, and Wf, respectively.

rectangles are located with high degrees of coincidences of ground truths. Compared with YOLOv3 (pruned, 3%), M-YOLOv3 (pruned, 15%) has similar performances on all these classes, with considerable results and slight mutual advantages of all these terms of criteria. YOLOv3 (pruned, 15%) has rather poor performances than other methods. The performance of M-YOLOv3 (pruned, 15%) only turn to worse on several examples and still competitive on the majority of these testing examples. The M-YOLOv3 has better performances than YOLOv3 with same pruning ratio 15%. The pruned version can be further pruned based on assigned pruning ratios, with certain loss of detecting

accuracy. The selection between the full version and pruned version of the modified YOLOv3 needs to be balanced in practical applications based on the requirements of efficiency and accuracy.

The examples of several typical detected defects are shown in Figure 2. The train YOLOv3 can deal with various kinds of surface defects, and the scale and location of the detected results are proper for the demonstrated samples.

3.2.2. Performance Comparisons on GC10-DET. In order to further verify the performances of the proposed M-YOLOv3 (full/pruned), the results on GC10-DET dataset are further compared with YOLOv3. With similar operations on DAGM dataset, the proposed M-YOLOv3 has also achieved better overall performance than the original YOLOv3 network on GC10-DET dataset. Their performance comparison on each type on GC10-DET has been demonstrated in Table 5, with IOU threshold of 0.5.

As shown in Table 5, the proposed M-YOLOv3 (full) has the best overall performances on all items. The proposed M-YOLOv3 (pruned) has considerable performances with YOLOv3 (full) and even better on these items of Recall, F1, and mAP. YOLOv3 (full) only has better performances slightly on several items for individual types and is poor than M-YOLOv3 (full/pruned) on the remaining results.

Since the GC10-DET dataset is more challengeable than DAGM dataset, the overall performances on these items are poor than these values on DAGM dataset. The GC10-DET dataset is collected in real applications, and these items has better persuasive for performance comparison. With overall consideration of effectiveness and efficiency, the proposed M-YOLOv3 (pruned) is more practical in most real applications. For time insensitive cases, the proposed M-YOLOv3 (full) is the best choice among them. The examples of several typical detected defects in GC10-DET dataset with M-YOLOv3 (pruned, 15%) are shown in Figure 3.

4. Discussion

In this paper, we adopt (pruned) M-YOLOv3 to deal with the topic of surface defects inspection. Compared with traditional processing methods with strategies of segmentation, the operating efficiency is improved significantly. Based on powerful processing abilities, various types of surface defects are detected by trained M-YOLOv3 with high accuracy. To further decrease operating efficiency, loss function optimization has been adopted with consideration of generalized intersection over union and area ratio. The pruning strategy is used to remove these branches of the YOLOv3 with less influence on its final outputs. The amount of parameters is reduced after a pruning strategy, with few impacts on the detecting accuracy.

Detected rectangles are useful in the purpose of locating the regions of surface defects. However, this representing method is still not considerately in detail, as too many normal pixels are included in the detected rectangles. The polygonal representation can be considered to modify the detected regions. Adversarial learning can be added in the training phase to deal with the issues of insufficient training examples and enhance the models' robustness on generalization ability. Besides, more effective pruning strategies can be designed to further decrease the number of parameters, with less influence on detecting accuracy.

Data Availability

The DAGM 2007 dataset can be download from the website https://conferences.mpi-inf.mpg.de/dagm/2007/prizes.html.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work is supported by Elite Plan of Shandong University of Science and Technology (No. 0104060540508), the Natural Science Foundation of Shandong Province under Grant (No. ZR2020MF132), and Research Funding of Post-Doctor who came to Shenzhen (No. E00120210001).

References

[1] Y. Lecun, Y. Bengio, and G. E. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[2] B. Jan, H. Farman, M. Khan et al., "Deep learning in big data analytics: a comparative study," *Computers & Electrical Engineering*, vol. 75, pp. 275–287, 2019.

[3] Z. Wang, S. Yoon, and D. S. Park, "Online adaptive multiple pedestrian tracking in monocular surveillance video," *Neural Computing and Applications*, vol. 28, no. S1, pp. 127–141, 2017.

[4] X. Xie, "A review of recent advances in surface defect detection using texture analysis techniques," *Electronic Letters on Computer Vision and Image Analysis*, vol. 7, no. 3, pp. 1–22, 2008.

[5] T. Czimmermann, G. Ciuti, M. Milazzo et al., "Visual-based defect detection and classification approaches for industrial applications—a survey," *Sensors*, vol. 20, no. 5, p. 1459, 2020.

[6] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *advances in neural information processing systems*, vol. 25, 2012.

[7] Z. Yu, X. Wu, and X. Gu, "Fully Convolutional Networks for Surface Defect Inspection in Industrial Environment," in *International Conference on Computer Vision Systems*, pp. 417–426, Shenzhen, China, 2017.

[8] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3431–3440, Boston, MA, USA, 2015.

[9] Cognex Vidi of CognexSeptember 2017https://www.cognex.cn/zh-cn/products/deep-learning/vidi-tools.

[10] Z. He and Q. Liu, "Deep regression neural network for industrial surface defect detection," *IEEE Access*, vol. 8, pp. 35583–35591, 2020.

[11] L. Qiu, X. Wu, and Z. Yu, "A high-efficiency fully convolutional networks for pixel-wise surface defect detection," *Access*, vol. 7, pp. 15884–15893, 2019.

[12] X. Wang, M. Yang, S. Zhu, and Y. Lin, "Regionlets for generic object detection," in *Proceedings of the IEEE international conference on computer vision*, pp. 17–24, Sydney, Australia, 2013.

[13] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Region-based convolutional networks for accurate object detection and segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 1, pp. 142–158, 2016.

[14] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE international conference on computer vision*, pp. 1440–1448, Santiago, Chile, 2015.

[15] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: towards real-time object detection with region proposal networks," *advances in neural information processing systems*, vol. 28, pp. 91–99, 2015.

[16] J. Deng, W. Dong, R. Socher, L. J. Li, K. Li, and L. Fei-Fei, "Imagenet: a large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255, Miami, FL, USA, 2009.

[17] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 779–788, Las Vegas, NV, USA, 2016.

[18] J. Redmon and A. Farhadi, "YOLO 9000: better, faster, stronger," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7263–7271, Honolulu, HI, USA, 2017.

[19] J. Redmon and A. Farhadi, "Yolov3: an incremental improvement," 2018, http://arxiv.org/abs/1804.02767.

[20] Z. Ge, S. Liu, F. Wang, Z. Li, and J. Sun, "Yolox: exceeding yolo series in 2021," 2021, http://arxiv.org/abs/2107.08430.

[21] H. Rezatofighi, N. Tsoi, J. Gwak, A. Sadeghian, I. Reid, and S. Savarese, "Generalized intersection over union: a metric and a loss for bounding box regression," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 658–666, Long Beach, CA, USA, 2019.

[22] Z. Cai, X. He, J. Sun, and N. Vasconcelos, "Deep learning with low precision by half-wave Gaussian quantization," in *Proceedings of the IEEE international conference on computer vision and pattern recognition*, pp. 5406–5414, Honolulu, HI, USA, 2017.

[23] H. Li, A. Kadav, I. Durdanovic, H. Samet, and H. P. Graf, "Pruning filters for efficient convnet," 2016, http://arxiv.org/abs/1608.08710.

[24] Z. Liu, J. Li, Z. Shen, G. Huang, S. Yan, and C. Zhang, "Learning efficient convolutional networks through network slimming," in *Proceedings of the IEEE international conference on computer vision*, Venice, Italy, 2017.

[25] Y. Cheng, D. Wang, P. Zhou, and T. Zhang, "Model compression and acceleration for deep neural networks: the principles, progress, and challenges," *IEEE Signal Processing Magazine*, vol. 35, no. 1, pp. 126–136, 2018.

[26] X. Lv, F. Duan, J. J. Jiang, X. Fu, and L. Gan, "Deep metallic surface defect detection: the new benchmark and detection network," *Sensors*, vol. 20, no. 6, p. 1562, 2020.

[27] M. Everingham, L. V. Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," *International Journal of Computer Vision*, vol. 88, no. 2, pp. 303–338, 2010.