한국빅데이터학회지
제7권 제2호, 2022, pp. 31-41 https://doi.org/10.36498/kbigdt.2022.7.2.31

Real-time Steel Surface Defects Detection Appliocation based on Yolov4 Model and Transfer Learning

Yolov4와 전이학습을 기반으로한 실시간 철강 표면 결함 검출 연구

김복경[1] · 배준희[1] · 환[2] · 이용은[2] · 옥영석[3†]

부경대학교 기술경영협동과정[1], (주)인타운[2], 부경대학교 기술경영전문대학원[3]

Abstract

Steel is one of the most fundamental components to mechanical industry. However, the quality of products are greatly impacted by the surface defects in the steel. Thus, researchers pay attention to the need for surface defects detector and the deep learning methods are the current trend of object detector. There are still limitations and rooms for improvements, for example, related works focus on developing the models but don't take into account real-time application with practical implication on industrial settings. In this paper, a real-time application of steel surface defects detection based on YOLOv4 is proposed. Firstly, as the aim of this work to deploying model on real-time application, we studied related works on this field, particularly focusing on one-stage detector and YOLO algorithm, which is one of the most famous algorithm for real-time object detectors. Secondly, using pre-trained Yolov4-Darknet platform models and transfer learning, we trained and test on the hot rolled steel defects open-source dataset NEU-DET. In our study, we applied our application with 4 types of typical defects of a steel surface, namely patches, pitted surface, inclusion and scratches. Thirdly, we evaluated YOLOv4 trained model real-time performance to deploying our system with accuracy of 87.1 % mAP@0.5 and over 60 fps with GPU processing.

■ Keyword : Surface Defects, Object Detection, YOLOv4, real-time detection

요 약

철강은 기계 산업의 가장 기본적인 구성 요소 중 하나이다. 그러나 철강의 표면 결함은 제품의 품질에 큰 영향을 미친다. 따라서 연구자들은 표면 결함 감지기의 필요성에 주목하고 딥 러닝을 이용한 방법은 객체 결함 감지를 하는데 많이 사용된다. 연구 개발용으로 학습 모델 개발에 초점을 맞추지만 실제 산업 환경에 실질적인 영향을 미치는 실시간 적용은 아직 적용되지 않는 한계와 개선의 여지가 필요하다.
본 연구는 YOLOv4를 기반으로 한 철강 표면 결함 감지의 실시간 적용을 제안한다. 첫째, 본 연구는 실시간 응용 모델을 적용하는 것을 목적으로 하며 실시간 객체 검출기의 가장 유명한 알고리즘 중 하나인 one-stage Detector의 YOLO 알고리즘을 중심으로 연구를 진행하였다. 둘째, 사전 훈련된 YOLOv4-Darknet 플랫폼 모델과 전이학습을 사용하여 철강 표면 오픈 소스 데이터셋 NEU-DET을 이용하여 학습과 테스트를 진행하였다. 본 연구에서는 철강 표면의 패치, 구멍 난 표면, 불순물, 스크래치 4가지 유형의 결함을 이용하였다. 셋째, 87.1% mAP@0.5의 정확도와 60fps 이상의 시스템 구축을 위해 YOLOv4를 이용하여 훈련된 모델의 실시간 성능을 평가하였다.

■ 중심어 : 표면 결함, 물체 검출, 욜로, 실시간 검출

2022년 11월 14일 접수; 2022년 12월 14일 수정본 접수; 2022년 12월 16일 게재 확정.

* This work was supported by the Technology Development Program(S3125098) funded by the Ministry of SMEs and Startups (MSS, Republic of Korea).

† 교신저자 (ysock@pknu.ac.kr)

Ⅰ. Introduction

In the process of industrial production, influences of many impacts e.g., equipment, technological processes, labor and site environment will lead to a variety of defects on the surface of product. Surface defects not only reduces the appearance quality and commercial value of the product but also affect the production performance and the safety and stability of subsequent steps in production line [1]. Therefore, surface defects detection has become an important task in factory production line. However, most detection task are completed by human, which has disadvantages of high labor cost and low efficiency. As a result, the demands of automated surface defects detection has drawn many attentions from researchers and company. Replacing human eyes with computer vision system has become a trend in industrial surface defect detection, which can be applied in many industrial files (steel, road, wood, fabric etc). With the booming development of deep learning based object detectors, defects can be classified as one of many labels in object detection. And defects detection based on deep learning methods are widely used in many industrial scenarios. However, the implementation of surface detection detections on embedded devices has two main issues. Firstly, surface defects of industrial products have variety of types and sizes, which leads to difficulty to not only detect and localize defects in a wide background area, but also classifies which kind of defects for further investigation. Secondly, most of research focused on improvement of the accuracy of detection but ignore the speed of detection, while the demand of online detection requires real-time performance with adequate accuracy.

From these problem statement, in this paper, we presents an implementation of high-accuracy and high-speed detection of surface defects based on YOLOv4. Firstly, we apply transfer learning with pre-trained YOLOv4 model on NEU-DET dataset, which contains steel surface defects.

Secondly, as NEU-DET dataset is small-scale dataset, we apply some traditional data augmentation techniques to improve generalization performance and prevent overfitting of model while training. Thirdly, we deploy YOLOv4 trained model on GUI application in order to evaluate real-time performance.

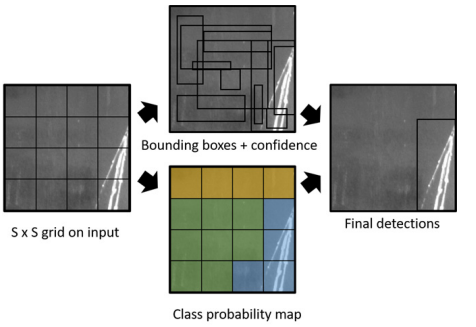
The study is organized as follows: Section 2 gives a summary of related work in this area. Section 3 illustrates the proposed methods, and Section 4 explains the experiments set up including all the datasets and data processing techniques and evaluation metrics. Section 5 shows the experiment results and evaluation analysis, followed by a discussion and further improvements. Section 6 gives a summary of our study in conclusion.

Ⅱ. Related Work

Due to the divergent emphasis on detection speed and detection accuracy, the object detection methods are gradually developed into two, including two-stage and one-stage method. One of the most representative method of a two-stage detectors is Faster R-CNN [2], which uses a region proposal network (RPN) to generate a region of interest (ROI) (or candidate box) and then achieve classification and position regression. Cheng et.al, [3] applied Faster Recurrent-Convolution Neural Network (Faster R-CNN) to damage detection of

drainage pipes, which achieved accuracy of 83 % mAP. Li and ..The authors in [4] adopted ZF-Net as the backbone of Faster R-CNN and added a max pooling layer at the head of the network to adapt defects of large-scale differences, which reached accuracy of 80.7% mAP. The trending method is one-stage detectors represented by Single-Shot Detection (SSD) [5] and You Only Look One (YOLO) [6], which uses a single-structure network to detect without RPN to generate a bounding box and achieves a higher detection speed. Zhang et al. [7] applied YOLOv3 with batch re-regularization and focal loss to detect bridge surface damage, which achieved a good performance. Yin et al. [8] used YOLOv3 to detect sewage pipeline damage defects and obtained accuracy of 85.37% mAP. Deng et al. [9] used YOLOv2 with graffiti interference to detect cracks and defects on a concrete surface under complex background. The authors also compared his YOLO method to RCNN in term of accuracy of 77% and 74.5% mAP respectively, and interference time of 0.17 s and 0.23 s respectively. Thus, the one-stage detector (particularly YOLO) is simpler and faster, which is more suitable for deploying real-time defect detection application. Besides, there are some previous works conducted experiments with NEU-DET dataset. The authors in [10] introduced classification priority with YOLOv3, which achieved the detection accuracy of 82.73% mAP and inference speed at 9.68 ms. In [11], the improved SSD with hard negative mining is proposed and achieved 72.4% mAP and detection speed of 27ms. From the literature review, the YOLOv3 [12], YOLOv2 [13], SSD, Faster R-CNN are quite out of date compared to YOLOv4. In previous, the authors didn't take into account deploy models into real application which can be practical implication to real industrial scenarios. In this work, we conduct experiments on NEU-DET dataset with pre-trained YOLOv4 using transfer learning and apply data augmentation techniques to improve generalization ability. In addition, we implement trained YOLOv4 model and deploy steel surface defect detector application with GUI to investigate the real-time performance on practical implication in industrial settings.

In this part, we will present briefly about YOLO history and focused on our trained YOLOv4. YOLO stands for You Only Look Once. YOLO is a state-of-the-art, real-time object detection system. It was developed by Joseph Redmon. It is a real-time object recognition system that can recognize multiple objects in a single frame. YOLO uses a totally different approach than other previous detection systems. It applies a single neural network to the full image. This network divides the image into regions and predicts bounding boxes and probabilities for each region. These bounding boxes are weighted by the predicted probabilities. The basic idea of YOLO is exhibited in the Figure 1 below [6]. YOLO divides the input image into an S × S grid and each grid cell is responsible for predicting the object centered in that grid cell.



<Figure 1> YOLO methodology

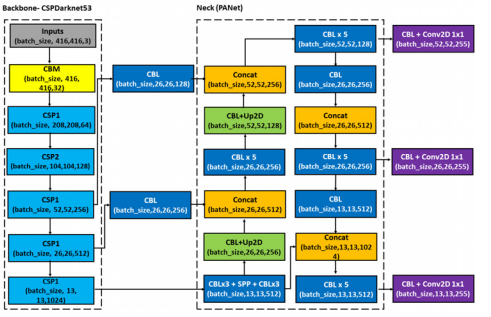
Each grid cell predicts B bounding boxes and confidence scores for those boxes. These confidence scores reflect how confident the model is that the box contains an object(Probability) and also how accurate it thinks the box is that it predicts(IOU). YOLO model has several advantages over classifier-based systems. It can recognize multiple objects in a single frame. It looks at the whole image at test time so its predictions are informed by the global context in the image. Also, it makes predictions with a single network evaluation unlike systems like R-CNN which require thousands for a single image. This makes it extremely fast, more than 1000x faster than R-CNN and 100x faster than Fast R-CNN. The YOLO design enables end-to-end training and real-time speeds while maintaining high average precision.

The most well-know YOLO iteration is YOLOv4 [14], which developed with Darknet framework [15] an Open Source Neural Networks in C.

Ⅲ. Methods

In this section, we will present our research methodology, which includes two phases namely training/testing phase and deployment phase. The first phase includes training and testing steps. Firstly, the open-source surface defects dataset will be collected and applied data augmentation techniques if necessary. Then, the selective dataset will be split into three parts, training, validation and testing set. Secondly, using transfer learning based on YOLOv4, we built, and tune the hyper-parameters of models on training set, and evaluates models performance on validation set. Finally, the trained models will be tested on test set, which is not include in training set, to prevent over-fitting of training models, and evaluate generalization performance. The second phase performs the task of deploying the best trained model on a GUI application. We will design a GUI template and generate executive application file to test our online application real-time performance. The detailed of our work will presented as follows.

3.1 YOLO-v4 Steel Surface Defects Detection Model Development



〈Figure 2〉 YOLOv4 architecture

YOLOv4 [14] is an object detection algorithm that is an evolution of the YOLOv3 model [12]. The YOLOv4 method is created by Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. It is twice as fast as EfficientDet with comparable performance. In addition, AP (Average Precision) and FPS (Frames Per Second) in YOLOv4 have increased by 10% and 12% respectively compared to YOLOv3. Its architecture is composed of CSPDarknet53 as a backbone, spatial pyramid pooling additional module, PANet path-aggregation neck, and YOLOv3 head as shown in Figure 2. YOLOv4 uses many new features and combines some of them to achieve state-of-the-art results: 65.7% AP50 for the MS COCO dataset at a real-time speed of~45.5 FPS

as shown in Table 1 [14].

〈Table 1〉 Comparison of methods

Method	AP50	FPS
YOLOv3_ASFF	63	45.5
RetinaNet	56.9	10.8
Faster R-CNN	59.2	9.4
SSD	48.5	22
Cascade R-CNN	62.1	8
YOLOv4	65.7	45.5

We train our model using YOLOv4-Darknet based on transfer learning. Thus, the pre-trained weights are downloaded from official Darknet website, and we train our models on custom dataset, particularly steel surface defects dataset. We try many trails with different batch-size, input network size (resolution) and model size to achieve the best trained models for our task. The details of training/testing setup will be present in Experiments section.

3.2 Deployment Developed Model into Application

Once the best trained model is obtained, we will deploy into application on various devices, including Ubuntu and Windows operating system. Firstly, we will design GUI template using PyQT5 platform with designer software in Anaconda. Secondly, the designer PyQt5 “.ui” file will be convert into python file “.py”, which will be load together with YOLOv4 best weights into application. Finally, the overall running source code and necessary files and libraries will be generated into single executive file with Pyinstaller software with Ubuntu, and Windows version.

Ⅳ. Experiments

4.1 Experiment Environment and Evaluation Metrics

As mentioned above, we use Darknet framework to build our YOLOv4 model. The whole training experiment is conducted on Anaconda virtual environment on the Ubuntu 18.04 64-bit as follows: Nvidia GeForce GPU RTX 2080Ti × 2, 11 GB Memory with Intel® Xeon(R) Silver 4110 CPU @ 2.10GHz × 16, 32 GB Memory, 500 GB Disk. In the experiment, the CUDA 11.0 backend and cuDNN 8.5.0 were used for GPU accelerations.

The mean Average Precision (mAP) is our main metric for evaluating model performance, which is the average of Average Precision (AP). AP is calculated by precision and recall, which is defined as the area under the P-R curve as follow:

$$AP = \int_0^1 p(r)dr \tag{1}$$

where: p(r) is Precision at Recall r.

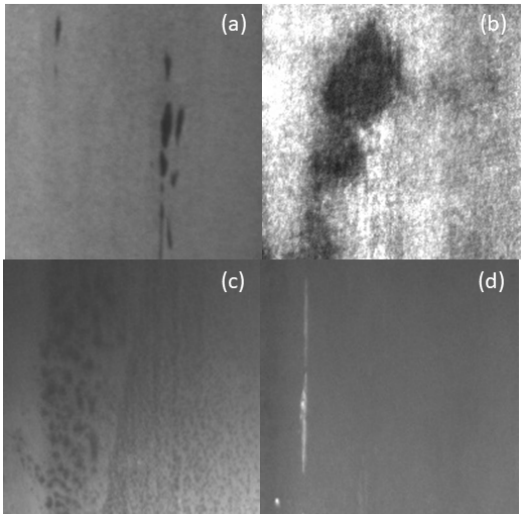
In addition, the mAP is the average of the calculated average precision of all classes. Thereby mAP is calculated as follows:

$$mAP = \frac{\sum_{i=1}^{K} AP_i}{K} \tag{2}$$

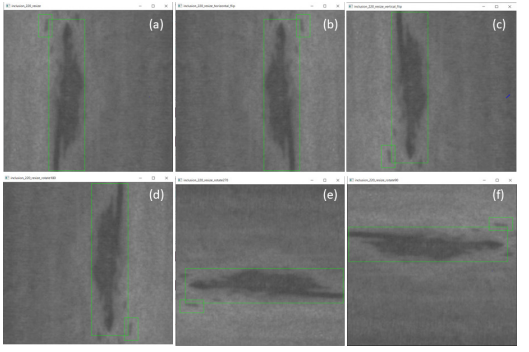
4.2 Dataset Preparation

NEU-DET dataset [16] is used in our experiments, which consists of hot rolled strip steel surface defects publicly released by Northeastern University. In our study, we uses four defects of

steel surface, namely inclusion, patches, pitted surface, and scratches for training. Figure 3 shows examples of the NEU-DET dataset. There are 300 samples for each defect, and a total of 1200 samples gray-scale images, with the original resolution of 200x200 pixels. For object detection task, the dataset provides bounding box annotations which are saved as an XML document, indicating the labels and bounding boxes of defects in each image.



〈Figure 3〉 Four type of defects in NEU–DET dataset, including (a) inclusion, (b) patches, (c) pitted surface, (d) scratches



〈Figure 4〉 Five traditional data augmentation techniques, (a) original, (b) horizontal flip, (c) vertical flip, (d) rotate 90, (f) rotate 180, and (g) rotate 270

We apply pre-processing method to original NEU-DET dataset, including resize original image size from 200x200 into 608x608, to increase size of small defects about 3 times, which leads to better detection performance on wide background area.

Also we utilize traditional data augmentation techniques, including rotate 90, 180, 270 angles, and flip horizontal, and flip vertical as shown in Figure 4, to increase the number of training images on small-scale dataset like NEU-DET. Thus, our data augmentation dataset includes totally 8400 images for 4 types of defect, or 2100 images for each defects. Then, we use cross-validation method to train our model on our data augmentation NEU-DET dataset, which avoid over-fitting when falling into local minima. Before the training, we divides our data augmentation NEU-Det dataset into train set, validation set, and test set in a ratio of 7:2:1, which contains 5880, 1680, and 840 images respectively.

4.3 Implementation Details

We set iteration to 8000 to start training with the pre-trained weighs file “yolov4.conv.137” and the initial hyper-parameter configuration on “yolov4-custom.cfg” file. The iterations (or max batches) is set to 8000 as the number of classes of 4 multiples with 2000 iterations per class. Others initial parameters of the training process are shown in Table 2.

〈Table 2〉 Initial Settings

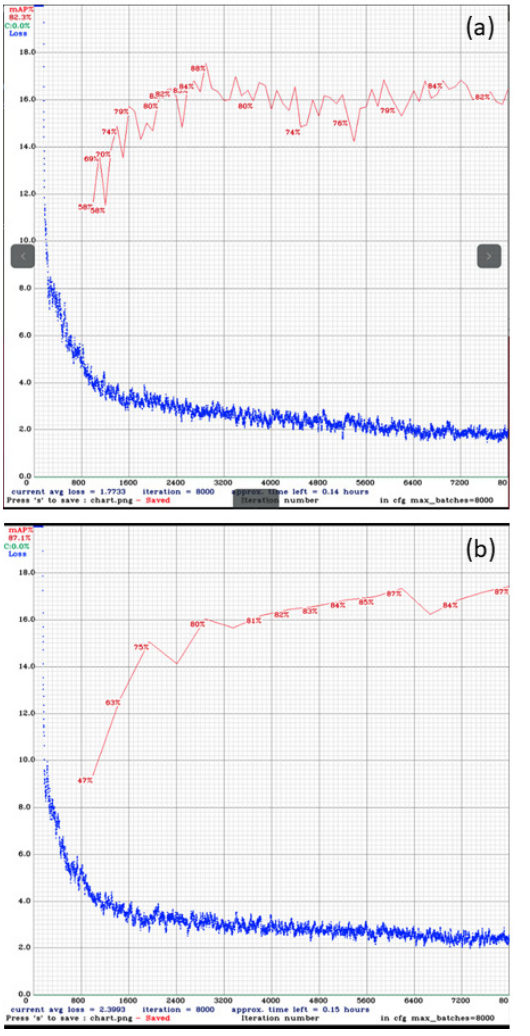
Parameters	Value
Size of input images	640x640
Batch & Subdivision	64 and 16
Momentum	0.949
Learning Rate	0.001

Apart from that, we customize the option of data augmentation operations (mix-up, random distort, random expand, random crop, random horizontal flip) during training process to improve the diversity of sample data and increase the generalization performance of our model.

4.4 Training/Testing Results

We train our model with pre-trained YOLOv4 weight on two datasets, including original and our data augmentation NEU-DET dataset. The training loss and accuracy mAP@0.5 during 8000 iterations are shown in Figure 5.

As can be seen in Figure 5(a), mAP@0.5 of our augmented dataset remains stable over 80% after reaching 2800 iterations, while mAP@0.5 of original dataset fluctuates greatly with the biggest gap of 12% between 88% and 76%. It means that our data augmentation dataset improves our trained YOLOv4 model performance on generalization ability. The highest mAP@0.5 while training with original and our data augmentation dataset achieves 88 and 87.1 % respectively as showed in Figure 5(a) and 5(b). Figure 6 shows the detail of testing results with 840 images, which is not included in training set, achieves mAP@0.5 of 87.11 %, precision of 0.89 and recall of 0.78 and F1-score of 0.83 at confidence threshold of 0.25.



〈Figure 5〉 Loss and mAP@0.5 during 8000 iterations of training, (a) original dataset and (b) our data augmentation dataset.

```
 calculation mAP (mean average precision)...
 Detection layer: 139 - type = 28
 Detection layer: 150 - type = 28
 Detection layer: 161 - type = 28
840
 detections_count = 7755, unique_truth_count = 2066
class_id = 0, name = inclusion, ap = 83.64%       (TP = 567, FP = 105)
class_id = 1, name = patches, ap = 93.15%         (TP = 529, FP = 33)
class_id = 2, name = pittedsurface, ap = 76.74%           (TP = 168, FP = 26)
class_id = 3, name = scratches, ap = 94.91%       (TP = 349, FP = 28)

 for conf_thresh = 0.25, precision = 0.89, recall = 0.78, F1-score = 0.83
 for conf_thresh = 0.25, TP = 1613, FP = 192, FN = 453, average IoU = 69.68 %

 IoU threshold = 50 %, used Area-Under-Curve for each unique Recall
 mean average precision (mAP@0.50) = 0.871108, or 87.11 %
Total Detection Time: 10 Seconds
```

〈Figure 6〉 Testing results with 840 images

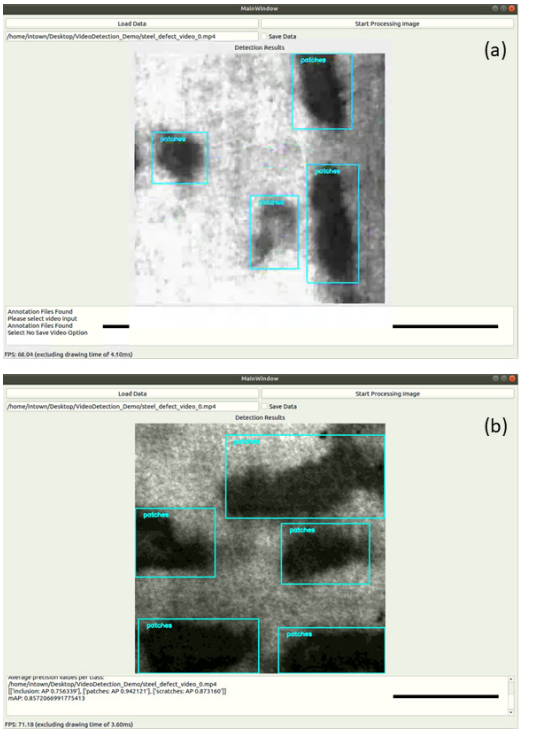
〈Table 3〉 Comparison with previous work

Methods	Accuracy mAP (%)	Inference Speed (ms)
YOLOv3 [10]	82.73	9.68 (GPU Tesla V100)
SSD [11]	72.4	27 (GPU RTX 2080Ti)
This work YOLOv4	87.11	16.66 (GPU RTX 2080Ti)

Table 3 presents the comparison between this work YOLOv4 and the previous studies YOLOv3 [10] and SSD [11]. It can be seen from Table 3 that this work YOLOv4 achieves higher accuracy mAP at 87.11 % compared to YOLOv3 and SSD with 82.73% and 72.4% respectively. The inference speed of this work is about 16.66ms (or 60 fps), which is faster than SSD method of 27ms on GPU RTX 2080Ti. In general, the model complixity of YOLOv4 is higher than YOLOv3, so the inference speed of YOLOv4 is higher than YOLOv3.

4.5 Development Application Testing Results

To test our steel surface defect detection application, we generates 5 videos from a set of shuffle 840 testing images with 30 fps. Figure 7 presents the GUI of our application, including 2 buttons “Load Data” and “Start Processing Image”, 1 selective option “Save Data”, 1 window screen to display “Detection Results”, and 1 screen to display “command” functions or some errors and detection time as frame per second (fps).. In details, the users can select different video input by clicking button “Load Data”, if the selected video has annotation files together with the video, then the command line will print out “Annotation Files Found” and “Please select video input” as in Figure 7(a) and after finished detection, the mAP@0.5 will be calculated as in Figure 7(b). If not, the application will only detect the video without calculating the mAP@0.5, since the annotation files are used to calculate mAP@0.5. Besides, if the user select the option “Save Data”, the video and frame by frame images of detection results will be saved for further investigation, and the command line will print out “Select Save Video Option”. Otherwise, the command line will print out “Select No save Video Option” as in Figure 7(a). When the user finished selecting input video and save data option, the application will be processed by clicking the “Start Processing Image”. Once each frame of video input (image) finished detection, the detection results will be displayed on window screen.



〈Figure 7〉 Our GUI application

The application demo is tested on computer with GPU, which achieved over 60 fps as shown in the bottom of Figure 8. The inference speed of the device should be at least between 30 and 60 frames per second (fps) in real industrial settings, where the distance between the camera and the production line is between 50 and 100 cm and the normal maximum speed of the running line is 30 meter per second (m/s) [17]. From experiments results, our application demo is achieved real-time performance on real industrial scenarios with 60ps. With the accuracy of more than 85% and inference speed of approximately 60 fps with advance GPU support, this application is applicable in industrial scenarios, which can support or replace the human force with strict supervision from expert.

Ⅴ. Conclusion

Nowadays, with the advanced of 4.0 industrial technology, instead of manual surface defects inspection, there is a requirement to automatically detect defects online or online monitoring in industrial production line. As a result, this paper presents a real-time surface defects detection application based on pre-trained YOLOv4 models and transfer learning. Experiments on the common dataset NEU-DET have obvious advantages in detection accuracy and interference speed. In details, mAP of four categories of steel surface defects achieves 87.1 % on our data augmentation NEU-DET and detection speed reaches over 60 fps, which achieves the aims of our research high precision and real-time application. In the future, instead of traditional augmentation techniques, more intelligent data augmentation methods, such as GAN, is consider to alleviate the problem of small-scale dataset of surface defects to further improve the overall performance and generalization abilities of the model.

참 고 문 헌

[1] Xing, J.; Jia, M. A convolutional neural network-based method for workpiece surface defect detection. Measurement 2021,176, 109185.

[2] S. Ren, K. He, R. Girshick and J. Sun, “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks” in IEEE Transactions on Pattern Analysis & Machine Intelligence, vol. 39, no. 06, pp. 1137-1149, 2017.

[3] Cheng, J.C.; Wang, M. Automated detection of sewer pipe defects in closed-circuit television images using deep learning techniques. Autom. Constr. 2018, 95, 155-171.

[4] Li, R.; Yuan, Y.; Zhang, W.; Yuan, Y. Unified vision-based methodology for simultaneous concrete defect detection and geolocalization. Comput.-Aided Civ. Infrastruct. Eng. 2018, 33, 527-544.

[5] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, “SSD: Single Shot MultiBox Detector,” Computer Vision – ECCV 2016. pp. 21-37.

[6] Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You only look once: Unified, real-time object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 26 June–1 July 2016; pp. 779-788.

[7] Zhang, C.; Chang, C.C.; Jamshidi, M. Bridge damage detection using a single-stage detector and field inspection images. arXiv 2018, arXiv:1812.10590

[8] Yin, X.; Chen, Y.; Bouferguene, A.; Zaman, H.; Al-Hussein, M.; Kurach, L. A deep learning-based framework for an automated defect detection system for sewer pipes. Autom. Constr. 2020, 109, 102967.

[9] Deng J, Lu Y, Lee VC-S. Imaging-based crack detection on concrete surfaces using You Only Look Once network. Structural Health Monitoring. 2021;20(2):484-499.

[10] Jiaqiao Zhang, Xin Kang, Hongjun Ni & Fuji Ren Surface defect detection of steel strips based on classification priority YOLOv3-dense network, Ironmaking & Steelmaking, 48:5, 547-558, DOI: 2021.

[11] Lv, X.; Duan, F.; Jiang, J.-j.; Fu, X.; Gan, L. Deep Metallic Surface Defect Detection: The New Benchmark and Detection Network. **2020**, 1562.

[12] Redmon, J.; Farhadi, A. Yolov3: An incremental improvement. arXiv 2018, arXiv:1804.02767.

[13] Joseph Redmon and Ali Farhadi. YOLO9000: Better, Faster, Stronger.,*CVPR2017*,.6517-6525, 2017.

[14] Bochkovskiy, A., C.-Y. Wang, and H.-Y.M. Liao YOLOv4: Optimal Speed and Accuracy of Object Detection. 2020. arXiv:2004.10934.

[15] YOLOv4 / Scaled-YOLOv4 / YOLO - Neural Networks for Object Detection (Windows and Linux version of Darknet). Available online: https://github.com/AlexeyAB/darknet (accessed on 24 Feb 2022).

[16] http://faculty.neu.edu.cn/yunhyan/NEU_surface_defect_database.html.(accessed on 24 Feb 2022).

[17] J. Li, Z. Su, J. Geng, and Y. Yin, “Real-time detection of steel strip surface defects based on improved YOLO detection network,” *IFAC-PapersOnLine*, vol. 51, no. 21, pp. 76-81, 2018.

저 자 소 개

김 복 경(Bok-Kyeong Kim)
·2000년 2월 : 동명대학교 정보통신공학과 (공학사)
·2013년 8월 : 부경대학교 기술경영협동과정 (공학석사)
·2015년 8월 : 부경대학교 기술경영협동과정 박사과정 수료
·2006년 3월～현재 : ㈜인타운 전무이사
<관심분야> : 빅데이터 활용, 기술경영

배 준 희(Jun-Hee Bae)
·1997년 2월 : 경일대학교 컴퓨터공학(공학사)
·2013년 8월 : 부경대학교 기술경영협동과정 (공학석사)
·2015년 8월 : 부경대학교 기술경영협동과정 박사과정 수료
·1997년 6월～현재 : ㈜인타운 대표이사
<관심분야> : 인공지능 활용, 스마트 제조

NGUYEN VIET HOAN
·2014년 7월 : 하노이과학기술대학교 전기통신공학 (공학사)
·2017년 8월 : 부경대학교 IT융합응용공학 (공학석사)
·2021년 9월～현재 : 부경대학교 인공지능 융합공학 박사과정
·2018년 1월～현재 : ㈜인타운 선임연구원
<관심분야> : 인공지능, 빅데이터

이 용 은(Yong-Eun Lee)

·2015년 2월 : 부산대학교 IT응용공학 (공학사)
·2017년 8월 : 서울대학교 전기정보공학 (공학석사)
·2022년 2월 : 부산대학교 기계 (공학박사)
·2017년 8월~현재 : ㈜인타운 책임연구원
<관심분야> : 스마트 제조, 기술경영

옥 영 석(Young Seok Ock)

·1993년 2월 : 한국과학기술원 산업공학 (공학박사)
·1987년 3월~2016년 2월 : 부경대학교 시스템경영공학부 교수
·2016년 3월~현재 : 부경대학교 기술경영전문대학원 교수