An End-to-end Steel Surface Defect Detection Approach via Fusing Multiple Hierarchical Features

Yu He, Kechen Song, Qinggang Meng, Yunhui Yan

***Abstract*—A complete defect detection task aims to achieve the specific class and precise location of each defect in an image, which makes it still challenging for applying this task in practice. The defect detection is a composite task of classification and location, leading to related methods are often hard to take into account the accuracy of both. And the implementation of defect detection depends on a special detection dataset which contains expensive manual annotations. In this paper, we proposed a novel defect detection system based on deep learning and focused on a practical industrial application: steel plate defect inspection. In order to achieve strong classification-ability, this system employs a baseline convolution neural network (CNN) to generate feature maps at each stage. And then the proposed multilevel-feature fusion network (MFN) combines multiple hierarchical features into one feature, which can include more location details of defects. Based on these multilevel features, a region proposal network (RPN) is adopted to generate regions of interest (ROIs). For each ROI, a detector, consisting of a classifier and a bounding box regressor, produces the final detection results. Finally, we set up a defect detection dataset NEU-DET for training and evaluating our method. On the NEU-DET, our method achieves 74.8/82.3 mAP with baseline networks ResNet34/50 by using 300 proposals. In addition, by using only 50 proposals, our method can detect at 20 fps on a single GPU and reach 92% of the above performance, hence the potential for real-time detection.**

***Index Terms*—Automated defect inspection, defect detection dataset (NEU-DET), defect detection network (DDN), multilevel-feature fusion network (MFN).**

I. Introduction

DEFECT inspection is a crucial step to guarantee the quality of industrial production, especially for steel plates. However, this process is usually performed manually in industry, which is unreliable and time-consuming. In order to replace the manual work, it is desirable to allow a machine to automatically inspect surface defects from steel plates with the use of computer vision technologies.

Fig. 1. Defect classification and defect detection task. (a) Defect classification task aims to "What", only outputting a defect class score. (b) Defect detection task aims to "What" and "Where", outputting a bounding box with a defect class score.

The founder of computer vision, British neurophysiologist Marr considers a vision task can be defined as "What is Where" that is the process of discovering what presents in an image and where is it [1]. Therefore, the object classification and detection are the most fundamental problems in the field of computer vision research [2]. Similarly, the automated defect inspection (ADI) can also be divided into two types: defect classification and defect detection. Given a defect image, the defect classification task is to solve if this image contains some class of defect (Fig.1a). And the defect detection task is to solve where a defect exists in this image, represented by a bounding box with a class score (Fig.1b). Therefore, *a complete defect detection task consists of two parts: defect classification, determining specific categories of defects, and defect localization, obtaining detailed regions of defects*. For defect inspection on steel plates, the detection task has superior advantages to complicated defects, e.g. multiple defects (Fig.2a), multiclass defects (Fig.2b) and overlapping defects (Fig.2c). The classification task can only find the defect with a highest category confidence in an image and not know the number of defects in Fig.2a, classes of defects in Fig.2b and emerge of an overlapping defect in Fig.2c. But for the follow-up quality assessment system, the quantity, category, and complexity of defects would be served as the chief indicators to evaluate the quality of a steel plate. It is apparent that defect detection can achieve a more comprehensive information reflection of a steel plate surface.

The previous ADI methods have two common problems: The one is the unclear usage of hand-craft features [3]-[5]. The

This work is supported by the National Natural Science Foundation of China (51805078, 51374063), the National Key Research and Development Program of China (2017YFB0304200), the Fundamental Research Funds for the Central Universities (N170304014, N150308001) and the China Scholarship Council (201806085007). (Corresponding authors: Kechen Song; Yunhui Yan)

Y. He, K. Song and Y. Yan are with the School of Mechanical Engineering and Automation, Northeastern University, Shenyang, 110819, China, and the Key Laboratory of Vibration and Control of Aero-Propulsion Systems Ministry of Education of China, Northeastern University, Shenyang, Liaoning, 110819, China.(e-mail:heyu142616@gmail.com,songkc@me.neu.edu.cn,yanyh@mail.neu.edu.cn).

Q. Meng is with the Department of Computer Science, LoughboroughUniversity, Loughborough LE11 3TU, U.K. (e-mail:, q.meng@lboro.ac.uk).

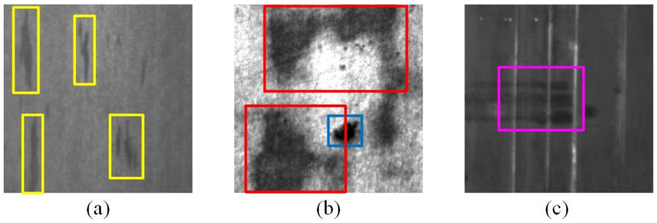


Fig. 2. Complicated defects. (a) Multiple defects. The yellow boxes indicate the defects belong to an identical class. (b) Multiclass defects. The red and blue boxes indicate the defects of different classes. (c) Overlapping defects. The pink box surrounds an overlapping region of defects of different classes.

determination of features is too subjective, and thereby human experience usually plays a decisive role in it. The other problem is imprecise defect localization (Fig.3a). Most methods only perform defect classification [6]-[8] or an incomplete defect detection. For example, some methods perform binary classification to find the regions of defects [9]-[10], or only provide a coarse region of a defect [11]-[12]. The recent developed Deep Learning (DL) technology can overcome the drawbacks of traditional ADI methods and have achieved significant results on many vision tasks. The DL can extract discriminative representations through a deep network (e.g. a CNN). These representations can reach a high level of abstract and therefore have strong representation-ability. The hand-craft features, by contrast, are merely the combination of low-level features [16]. Moreover, DL can train on location-annotated samples to obtain precise location information.

At present some studies have already applied DL for ADI. However, most methods can only perform defect classification due to the lack of special datasets [18]-[21]. The defect classification seems to be oversimplify and unable to provide location information. Other methods use a combination of DL and traditional image processing to perform defect detection or segmentation [17]. These methods always use a DL classifier in parallel with a detector or a segmenter that based on traditional image processing. This way can eliminate the need for special training datasets but damage the end-to-end characteristic of DL system and lose the intelligence and generalization to some extent. Unlike the above-mentioned methods, we attempt to establish an end-to-end defect detection system for ADI, which can provide a bounding box with a class score for precisely classifying and locating a defect (Fig.3b). A DL-based segmenter like Mask R-CNN [13] seems to be better for showing the shape of a defect. However, this kind of segmenter will consume huge amounts of computation source which cannot meet the real time demand of industrial inspection. Furthermore, it is highly impracticable for industry to build a large instance-level defect segmentation dataset, and thereby this kind of segmenter is almost impossible to apply. So, it is the best trade-off to perform defect detection for ADI at present.

This paper mainly addresses three challenges. First, the detection system needs strong classification-ability. The common classification problems such as inter-class similarity, intra-class difference and background interference are also present in ADI [9] [11]. Therefore, we equip a deep network ResNet into the system as the backbone [23]. As current research in transfer learning [15], the key to drive large networks is pretraining on ImageNet [22]. The detection system can gain strong classification power by training ResNet on enough data.

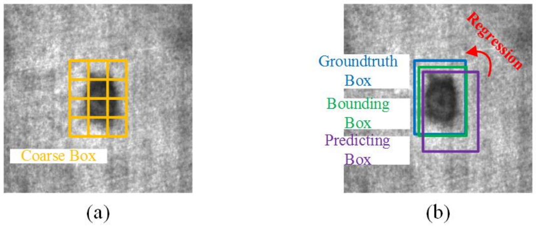


Fig. 3. Different styles of obtaining a defect region. (a) Many previous detectors based on hand-craft features directly combine related spatial cells into a block through various special approaches. The block is regarded as a detection region, which is a coarse box without refining. (b) Detectors based on DL mainly use regression methods to refine a predicting box. Through a large amount of iterative learning, the predicting box is gradually close to the groundtruth box. Finally, the refined box is regarded as the bounding box of the defect, which can represent the precise location information of the defect.

Second, the challenge of performing defect localization using CNN features in DL-based methods remains. As we known, the convolutional layers of CNN can be regarded as filters, which results in some location details will be gradually lost when an image flows in the CNN. Usually, DL-based methods perform localization based on the last convolutional feature map [14] [28] [34]. Our method is to fuse multiple feature maps. Because the feature maps exhibit diverse characteristics at each stage of CNNs: the shallow features have rich information but not discriminative enough, and the deep features are semantic robustly but lose too many details. In other fields [34], the Hypernet also uses more features but they are mainly selected from the latter part of the network. The proposed MFN combines the multiple features covering all stages. We address the detection from the industrial perspective. Since grey images have less information than RGB images, the MFN must include lower-level features which are discarded by HyperNet. Furthermore, the MFN uniforms the size of multiple features before fusion which can not only save more details of images but also use less parameters of models.

Third, in defect detection, data annotation is expensive, because one has to draw a defect's bounding box and assign a class label to it. Recent progress in this field can be attributed to two factors: 1) ImageNet pretrained models and 2) large baseline CNNs, which made great progress in DL-based defect classification [18]-[20]. But the limited data and expensive annotation still limit the development of defect detection. In this paper, we open a defect detection dataset NEU-DET for fine-tuning models. When the DL models have finished training on a special dataset, they can be used to perform the defect detection task.

This paper establish an end-to-end ADI system, called Defect Detection Network (DDN), in an attempt to overcome the above-mentioned challenges. The DDN 1) adopts a strong ResNet in defect classification, 2) proposes the multilevel-feature fusion network (MFN) to assemble more location details, and 3) sets up a defect detection dataset for fine-tuning and reports improvements on it. In more details, first, we pretrain the ResNet on the ImageNet and fine-tune all the models on the NEU-DET. The MFN can fuse the selected features into a multi-level feature which has characteristics covering all the stages of the ResNet. Next, a RPN is adopted in proposals generation based on the multi-level features and then the DDN can output the class scores and the coordinates of bounding-box. Finally, we evaluate the proposed method on NEU-DET and the results can demonstrate a clear superior to other ADI methods.

To summarize, the main contributions of this paper are:

- the introduction of the end-to-end defect detection pipeline DDN that integrates the ResNet and the RPN for precise defect classification and localization;
- the proposed MFN for fusing multi-level features. Compared with other fusing methods, MFN can combine the lower-level and higher-level features, which makes multi-level features have more comprehensive characteristics; and
- a defect detection dataset NEU-DET for fine-tuning networks and a demonstration that the proposed DDN has a very competitive performance on this dataset.

II. Related Work

A. Defect Inspection

Generally, a defect classification method includes two parts: a feature extractor and a classifier. The classic feature extractor is to obtain hand-craft features such as HOG and LBP, and they are always followed by a classifier e.g. SVM. Therefore, the combination of different feature extractors and classifiers produces a variety of defect classification methods. For instance, Song *et al.* [3] improves the LBP to against noise and adopt NNC and SVM to classify defects. Ghorai *et al.* [9] is based on a small set of wavelet features and uses SVM to perform defect classification. Different from above mentioned two methods, Chu *et al* [8] employs a general feature extractor and enhance SVM. From the perspective of computer vision, the defect classification task is essentially defect image classification, which is struggled in complicated defect images. To solve it, the simple and direct way is to perform defect localization before defect classification making the inspection task classify on regions of defects instead of a whole defect image, which is the defect detection task. For example, the defect detectors in [11] and [12] firstly perform a 0-1 classification to judge features whether belong to a defect class or a nondefect class, and then finds defect regions based on the boundary of defect-class features, finally perform different classification methods to determine the specific class of a defect. Besides, there is another simplified detector for the requirement of quick detection, which only focuses on regions of defects but regardless of the defects are in different categories, e.g. [10].

However, the DL-based methods differ radically from the above methods. Hand-craft feature extractor locally analyses a single image and extract features. But CNN is to construct the representation of all the input data through a large amount of learning. CNN has fine generalization and transferability so that there are some defect inspection methods based on CNN. For example, Chen *et al* [21] demonstrates that an object detector like Overfeat [24] can be transferred to be a defect detector by some means. And like [18] and [19], they demonstrate that using a sequential CNN to extract features can improve classification accuracy on defect inspection. Similarly, based on a sequential CNN, Ren *et al* [17] performs an extra defect segmentation task on classification results to define the boundary of a defect. Moreover, Natarajan *et al.* [20] employs a deeper neural network VGG19 for defect classification. With the depth of CNN, the defect classification accuracy has been further improved.

B. Baseline Networks

There are three popular CNN architectures at present, which are used as baseline networks for pretraining. The early successful networks are based on the sequential pipeline architecture [25], which establish the basic structure of CNN and prove the importance of depth of networks. Subsequently, the Inception networks employed modular units which increase both the depth and width of a network without the increment of computational cost [26]. The third type is ResNet using residual blocks to make networks deeper without overfitting [23]. ResNet is widely applied in various vision tasks, achieving competitive results with a few parameters.

Choosing a proper baseline network is the key to gain good results for DL methods. A large network has strong represent-ability for input data hence the extracted features at high-abstract level, but there is a great demand for training data.

C. CNN Detectors

The CNN detectors aim to classify and locate each target with a bounding box. They are mainly divided into two methods: one is the region-based method and another is the direct regression method. The most famous region-based detectors are the “R-CNN family” [27] [28] [14]. In this framework, thousands of class-independent region proposals are employed for detection. Region-based methods are superior in precision but require slightly more computation. The representative direct regression methods are YOLO [29] and SSD [30]. They directly divide an image into small grids and then for each grid predict bounding boxes which then regressed to the groundtruth boxes. The direct regression method is fast to detect but struggles in small instances.

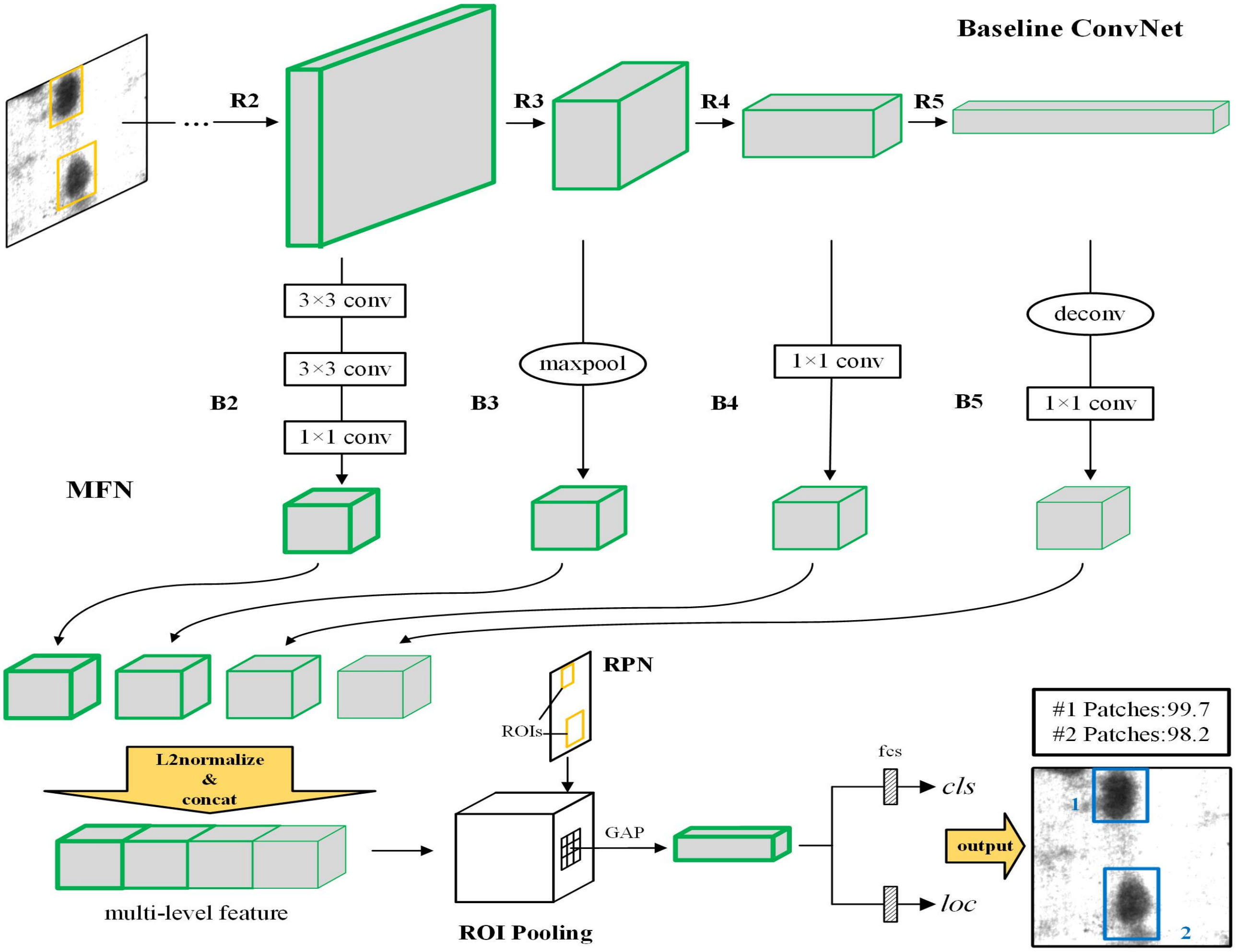


Fig. 4. Defect Detection Network (DDN).In a single pass, we extract features from each stage of the Baseline ConvNet which then fused into a multilevel feature by MFN. RPN is adopted to generate regions of interest (ROIs) based on the multilevel feature. For each ROI, the corresponding multilevel feature is transformed into a fixed-length feature through the ROI pooling and the global average pooling (GAP) layers. Two fully-connected (fc) layers process each fixed-length feature and feed into output layers producing two results: a one-of-(C+1) defect class prediction (*cls*) and a refined bounding box coordinate (*loc*).

III. Defect Detection Network

In this section, the defect detection network (DDN for short) is described in detail (see Fig.4). A single-scale image of an arbitrary size is processed by a CNN, and the convolutional feature maps at each stage of the ConvNet are produced (ConvNet represents the convolutional part of a CNN). We extract multiple feature maps and then aggregate them in the same dimension by using a lightweight multilevel-feature fusion network, named MFN. In this way, MFN features have the characteristics from several hierarchical levels of ConvNet. Next, Region Proposal Network (RPN) [14] is employed to generate region proposals (ROIs) over the MFN feature. Finally, the MFN feature corresponding to each ROI is transformed into a fixed-length feature through the ROI pooling [28] and the global average pooling (GAP) layers. The feature is fed into two fully-connected layers. One is a one-of-(C+1) defect classification layer ("*cls*") and the other is a bounding-box regression layer ("*loc*").

The rest of this section introduces the details of DDN and motivates why we need to design MFN into the network for the defect detection task.

A. Baseline ConvNet Architecture

As we know that pretraining on the ImageNet dataset is important to achieve competitive performance. And then this pretrained model can be fine-tuned on a relatively small defect dataset. In this paper, we select the recent successful baseline network ResNet as the backbone. ResNet presents several attractive advantages: a) ResNet can achieve state-of-the-art precision with extremely few parameters, in comparison with the CNN of sequential pipeline architecture of the same magnitude (ResNet50 vs. VGG16, 0.85M vs. 138M parameters). It implies that ResNet has lower computational cost and less probability of overfitting. b) ResNet uses GAP to process the final convolutional feature map instead of the dual stacked fully-connected layers, which can be in a manner of preserving more comprehensive location information of defects in the image. c) ResNet has a modularized ConvNet, which is easy to integrate.

In this paper, we select ResNet34 and ResNet50 as baseline networks. The detailed structures of both networks are shown

TABLE I
ARCHITECTURE OF BASELINE NETWORKS

Block name	Conv1	R2	R3	R4	R5	Output
ResNet34	7×7,64, stride=2	$\begin{bmatrix}3\times3,64\\3\times3,64\end{bmatrix}\times3$	$\begin{bmatrix}3\times3,128\\3\times3,128\end{bmatrix}\times4$	$\begin{bmatrix}3\times3,256\\3\times3,256\end{bmatrix}\times6$	$\begin{bmatrix}3\times3,512\\3\times3,512\end{bmatrix}\times3$	GAP Fc Softmax
ResNet50		$\begin{bmatrix}1\times1,64\\3\times3,64\\1\times1,256\end{bmatrix}\times3$	$\begin{bmatrix}1\times1,128\\3\times3,128\\1\times1,512\end{bmatrix}\times4$	$\begin{bmatrix}1\times1,256\\3\times3,256\\1\times1,1024\end{bmatrix}\times6$	$\begin{bmatrix}1\times1,512\\3\times3,512\\1\times1,2048\end{bmatrix}\times3$	
Output size	112×112	56×56	28×28	14×14	7×7	1×1

in Table I, and residual blocks are denoted as {R2, R3, R4, R5}.

B. Produce Multilevel Features

Previous excellent approaches only utilize high-level features to extract region proposals (like the Faster R-CNN extract proposals upon the last convolutional feature maps). In order to obtain quality region proposals, single-level features should be extended to multilevel features. Obviously, the simplest way is to assemble feature maps from multiple layers [31]. So now comes the question, which layers should be combined? There are two essential conditions: non-adjacent, because adjacent layers have highly local correlation [32], and coverage, including features from low-level to high-level. For a ResNet, the most intuitive way is to combine the last layers in each residual block.

To fuse features at different levels, the proposed network MFN is appended on the pretrained model. MFN has four branches, denoted as {B2, B3, B4, B5}, and each branch is a small network. B2, B3, B4, and B5 are sequentially connected to the last layer of R2, R3, R4, and R5. When an image flows through the baseline ConvNet, the Ri features are produced in order. The Ri feature means the feature map output from the last layer of the residual block Ri, i = 2,…,5. Similarly, the Bi feature is the feature map produced from the last layer of the MFN batch Bi, i = 2,…,5. And then each of Ri features is led to the corresponding branch in MFN producing Bi features. Finally, multilevel features are obtained via concatenating the B2, B3, B4, and B5 features, which come from different stages of a CNN.

As a final note, MFN is efficient in computation and strong in generalization. MFN can reduce required parameters via modifying the number of filters of 1×1 conv. This operation may hurt accuracy but prevent overfitting in the case of insufficient training data.

C. Extract Region Proposals

The region proposal network (RPN) is employed to extract region proposals by sliding on the multilevel feature maps. RPN takes an image of arbitrary size as input and outputs anchor boxes (candidate boxes), each with a score representing whether it is a defect or not. The originality of RPN is the "anchor" scheme that makes anchor boxes in multiple scales and aspect ratios. And then anchor boxes are hierarchically mapped to the input image so that region proposals of multiple scales and aspect ratios produced. As a result of the resolution size of MFN feature, the RPN can be considered as sliding on the R4 feature. Follow [14], we set 3 aspect ratios {1:1, 1:2, 2:1}. Considering the multiple sizes of defects, we set 4 scales {64^2, 128^2, 256^2, 512^2}. So RPN produces 12 anchor boxes at each sliding location.

The region proposal extractor always ends with a ROI pooling layer. This layer performs max-pooling operation over a feature map inside each ROI to convert it into a small feature vector (512-d for ResNet34 and 2048-d for ResNet50) with a fixed size of W×H (In this paper, 7×7). At last, based on these small cubes, calculate the offset of each region proposal with an adjacent groundtruth box and the probability whether there exist defects.

For a single image, RPN may extract thousands of region proposals. To deal with the redundant information, the greedy non-maximum suppression (NMS) is often applied for eliminating high-overlap region proposals. We set the IOU threshold for NMS at 0.7, which can discard a majority of region proposals. After NMS, the top-K ranked region proposals are selected from the rest. In the following we fine-tune DDN using top-300 region proposals owing to the extracted quality region proposals, but reduce this number to accelerate the detection speed without harming accuracy at test-time.

IV. TRAINING

A. Multitask Loss Function

The defect detection task can be divided into two subtasks, hence DDN has two output layers. The *cls* layer outputs a discrete probability distribution, $k = (k_1,...,k_C)$, for each ROI over $C + 1$ categories (C defect categories plus one background category). As usual, k is computed by a softmax function. The *cls* loss L_{cls} is a log loss over two classes (defect or not defect). $L_{cls} = -\ log\ (k, k^*)$ where k^* is the groundtruth class. And the *loc* layer outputs bounding box regression offsets, $t= (t_x, t_y, t_w, t_h)$, for each of the C defect categories. As [28], the *loc* loss L_{loc} is a smooth L1 loss function. $L_{loc} = SmoothL1\ (t- t^*)$ where t^* is the groundtruth box associated with a positive sample. For bounding box regression, we adopt the parameterizations of t and t^* given in [27]:

$$\begin{aligned} & t_x = (x - x_a)/w_a, \quad t_y = (y - y_a)/h_a, \\ & t_w = \log(w/w_a), \quad t_h = \log(h/h_a), \\ & t_x^* = (x^* - x_a)/w_a, \quad t_y^* = (y^* - y_a)/h_a, \\ & t_w^* = \log(w^*/w_a), \quad t_h^* = \log(h^*/h_a). \end{aligned} \tag{1}$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIM.2019.2915404, IEEE Transactions on Instrumentation and Measurement

Where the subscripts x, y, w, and h denote each box's center coordinates and its width and height. The variables x, x_a and x^* separately represent the predicted box, anchor box, and groundtruth box (the same rules for y, w, and h).

With these definitions, we minimize a multitask loss function, which is defined as:

$$L(k,k^*,t,t^*) = L_{cls}(k,k^*)+\lambda p^* L_{cls}(t,t^*) \quad (2)$$

Where λ is the weight parameter balancing both *cls* and *loc* terms. During training, we set $\lambda = 2$ indicating that DDN is devoted to achieving better defect locations. p^* is the activation parameter of the *loc* term. The localization loss is involved in the subsequent calculation only for positive samples (p^*=1) and is disabled otherwise (p^*=0). We follow the "IOU" strategy in [14] to determine the positive and negative samples from anchors.

B. Joint Training

For pretrained network, MFN and RPN are new layers. Hence we need to make these three networks share the common convolutional features through training. The pretrained model essentially is a classification network, and multilevel features generated from MFN can be directly fed into the *cls* layer. So the pretrained network and MFN can be merged into one network, and then performed an end-to-end training. Without RPN, the rest of DDN is a detector network. To share features with RPN, the 4-step alternating training strategy in [14] is adopted, alternating between training RPN and training detector network. Combining these two strategies, we develop a practicable 5-step joint training algorithm, which is shown in Algorithm 1.

Algorithm 1 5-step Joint Training Algorithm

```
Input:   Defect images with annotations
Step1:   Train the merged network for initializing MFN
         with the pretrained model, obtaining model $M_P$.
Step2:   Train RPN based on $M_P$, generating proposals $P^*$.
Step3:   Train the detector network using proposals $P^*$,
         obtaining model $M_D^*$.
Step4:   Fine-tune RPN based on $M_D^*$, generating
         proposals $P$ and obtaining model $M_R$.
Step5:   Fine-tune the detector network using proposals
         $P$, obtaining model $M_D$.
Output:  Combine $M_R$ and $M_D$ as the final model.
```

After step2 and step3, RPN and the detector network are initialized with the ImageNet pretrained model in succession. However, these two networks have not shared the convolutional features at this point. They get it until the fine-tuning processes of step3 and step4 are finished. Specifically, we freeze the shared convolutional layers and only fine-tune the unshared layers. Finally, we combine two networks as a united network.

C. Implementation

For DDN, we adopt image-centric training strategy. Images are resized such that their short side is 600 pixels. We use SGD to train with a weight decay of 0.0001 and a momentum of 0.9. We take a single image per mini-batch iteration. The mini-batch size is 64 for detector network training (include MFN training), and 128 for RPN training. We fine-tune the model using a learning rate of 0.001 for 200k mini-batch iterations and 0.0001 for another 100k mini-batch iterations. We use "Xavier" initialization for all new layers [33]. To avoid overfitting we also use several data augmentation methods such as rotation, reflection, shift etc., but remove the dropout module.

V. Experiments

The performance of DDN is evaluated on our defect datasets: NEU-CLS and NEU-DET. We demonstrate that DDN achieves a reasonable design and promising results.

A. NEU-DET Dataset

NEU surface defect[1] is a defect classification dataset that we opened seven years ago [3]. There are six types of defects from hot-rolled steel plates, including crazing, inclusion, patches, pitted surface, rolled-in scales and scratches. Each class has 300 images, but it does not mean that an image consists of a single defect. Examples of defect images are shown in Fig. 5.

To perform defect detection tasks, we provide annotations saved as XML files. With them, the classification dataset is upgraded to a detection dataset. The annotation marks the class and bounding box of each defect appearing in an image. Each bounding box is regarded as a groundtruth box, which is represented by its top left and bottom right coordinates. There are nearly 5000 groundtruth boxes in total. For simplicity, we call the original dataset NEU-CLS, and the complemented

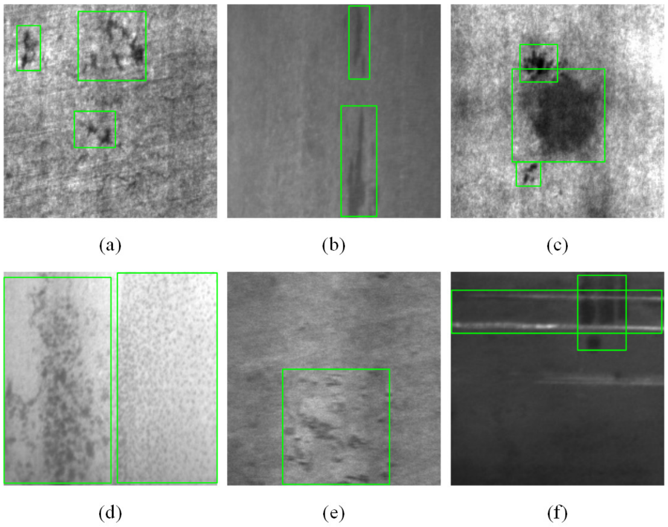


Fig. 5. Examples of defect images with annotations in NEU. The green box is a groundtruth box which has a class label and two corner coordinates of the box (top-left and bottom-right). The category to which the image belongs: (a) Crazing. (b) Inclusion. (c) Patches. (d) Pitted surface. (e) Rolled-in scale. (f) Scratches.

[1] The NEU dataset has been introduced in our previous work [3]. If you want to know the details about the datasets, please visit the website: http ://faculty.neu.edu.cn/yunhyan/NEU_surface_defect_database.html

TABLE II
DETECTION RESULTS ON NEU-DET

Method	Network	mAP	crazing	inclusion	patches	pitted surface	rolled-in scale	scratches
FRCN	VGG16	72.3	42.9	67.9	84.9	79.1	68.8	89.9
HyperNet	VGG16	74.8	54.1	68.0	86.5	87.0	65.2	88.1
DDN	VGG16	76.6	50.8	71.2	90.7	88.5	69.0	89.3
FRCN	ResNet34	70.2	46.7	61.3	82.8	76.5	70.7	83.4
FRCN	ResNet50	77.9	52.5	76.5	89.0	84.7	74.4	90.3
DDN	ResNet34	74.8	48.0	75.9	87.4	78.3	68.4	90.8
DDN	ResNet50	82.3	62.4	84.7	90.7	89.7	76.3	90.1

dataset NEU-DET. Examples of annotations are also shown in Fig. 5.

B. *Defect Classification on NEU-CLS*

As mentioned above, MFN can be merged into baseline CNNs for defect classification tasks. Therefore, we first report results on defect classification to demonstrate that our approach can achieve the competitive accuracy over other related methods. And merging MFN does not significantly affect the classification ability. Fig. 6 shows the defect classification results compared with other methods. According to Fig. 6, we can get the following conclusions:

1) The networks with MFN can perform well on defect classification so the multilevel features still have strongly semantical capability.

2) For ResNet34, MFN slightly harms the classification results. But this influence is vanished for the deeper network ResNet50. It indicates that features extracted from deeper network are more distinctive hence the entire network becomes more robust.

3) With MFN, the ResNet34 obtain 99% of the accuracy of the ResNet50, which indicates that in practice a very deep network is not really required for defect classification task.

As we know, stronger performance on defect classification should be positively correlated with stronger performance on defect detection. A good classification result is the prerequisite for subsequent defect detection experiments.

C. *Defect Detection on NEU-DET*

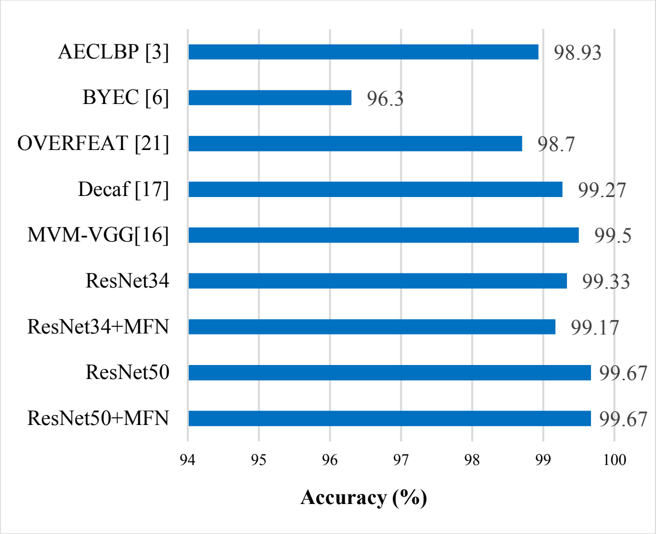


Fig. 6. Classification results on NEU-CLS dataset.

We carry out defect detection experiments on NEU-DET dataset. Conventionally, we divide the NEU-DET into training set and test set, and fix the training/testing split. The training set containing 1260 images used for fine tuning the network introduced in section IV-B, and the test set containing 540 images. We compare DDN with Faster R-CNN and HyperNet [34] on the test set and both methods use the same baseline network (VGG16 [40]) mentioned in their papers. In addition, DDN and Faster R-CNN are also experimented on ResNet34/50 due to the similar proposals generator. Unlike defect classification, only accuracy is not an appropriate performance measure in case of defect detection. Therefore, we evaluate the results of detection experiments by average precision (AP) which is a good trade-off between two significant detection indexes: Precision and Recall. These indexes are defined as follows:

$$\text{Precision}=\frac{\text{TP}}{\text{TP+FP}} \tag{3}$$

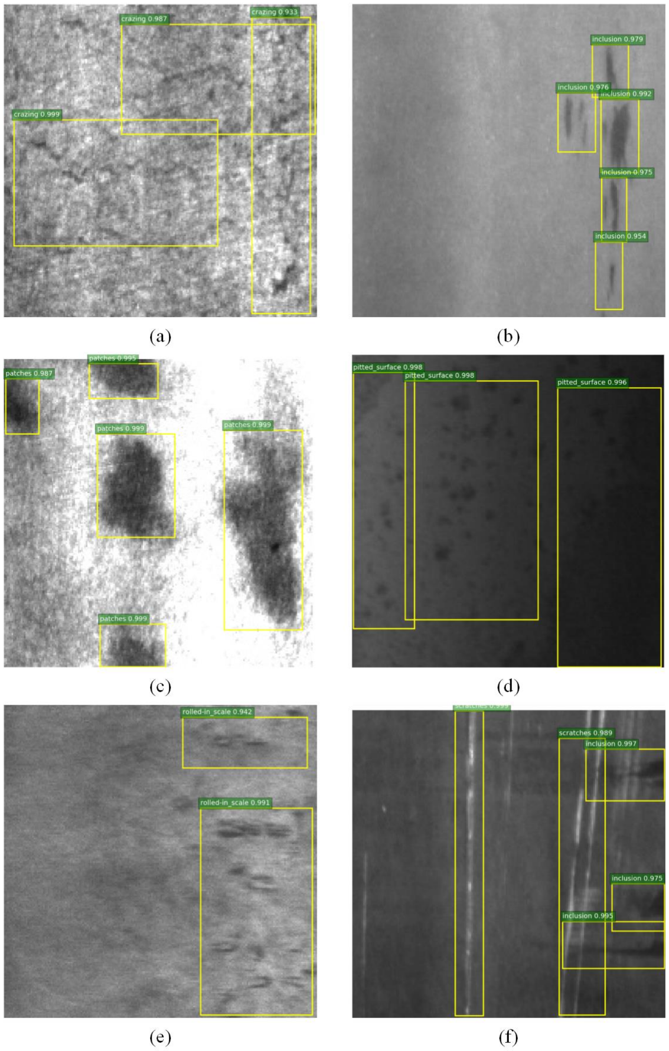
$$\text{Recall}=\frac{\text{TP}}{\text{TP+FN}} \tag{4}$$

$$\text{AP}=\frac{\text{Precision+Recall}}{2} \tag{5}$$

Where TP, FP, and FN represent the number of true positives, false positives, and false negatives, respectively. And the mean average precision (mAP) is also calculated to evaluate the overall performance, which is the mean value of the AP of all the classes.

Table II shows the results of defect detection experiments. Under the baseline ResNet34/50, DNN achieves a mAP of 74.8/82.3, 4.6/4.4 higher than Faster R-CNN. This result demonstrates the proposals extracted from multilevel features are superior to the proposals extracted from single-level features. Under the same baseline network (VGG16), Faster R-CNN achieves a mAP of 72.3 and HyperNet achieves a mAP of 74.8. DNN achieves a mAP of 76.6, 4.3 points higher than Faster R-CNN and 1.8 points higher than HyperNet. HyperNet is also a detector based on the multiple features, but our method can extract higher quality region proposals, which will be discussed in Section VI in details. The examples of detection results on NEU-DET are showed in Fig. 7.

Through the previous defect classification experiments, it is proved that MFN effects slightly on classification accuracy. Therefore, the improvement of mAP is benefited from the quality region proposals extracted from multilevel features. That means MFN contributes to improve the localization accuracy. We specifically evaluate the performance of MFN in the next section.



(a) (b)

(c) (d)

(e) (f)

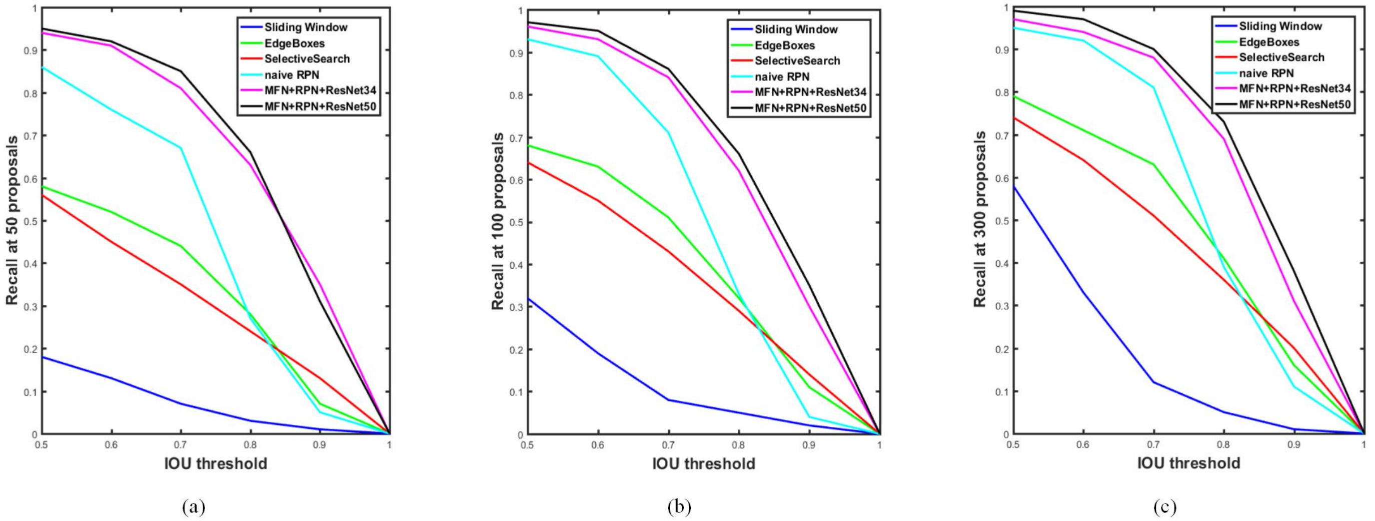
Fig. 7. Examples of detection results on NEU-DET. For each defect, the yellow box is the bounding box indicating its location and the green label is the class score. The subset to which the image belongs: (a) crazing (b) inclusion (c) patches (d) pitted surface (e) rolled-in scale (f) scratches.

D. Analysis on MFN

To verify MFN is able to improve localization accuracy, we compare with several region proposal extractors, sliding window, Edge Boxes [35], and Selective Search [36]. Besides these methods, RPN+MFN is also compared with the naive RPN (extract proposals based on single-level features). If the quality of proposals gets improved, the detector can use fewer proposals and stricter IOU thresholds without harming recall. Therefore, we evaluate recall on NEU-DET test set with different numbers of proposals and IOU thresholds. The number of proposals is the top-K ranked region proposals selected by these methods. IOU denotes a ratio between intersection and union of the predicted boxes and the groundtruth boxes.

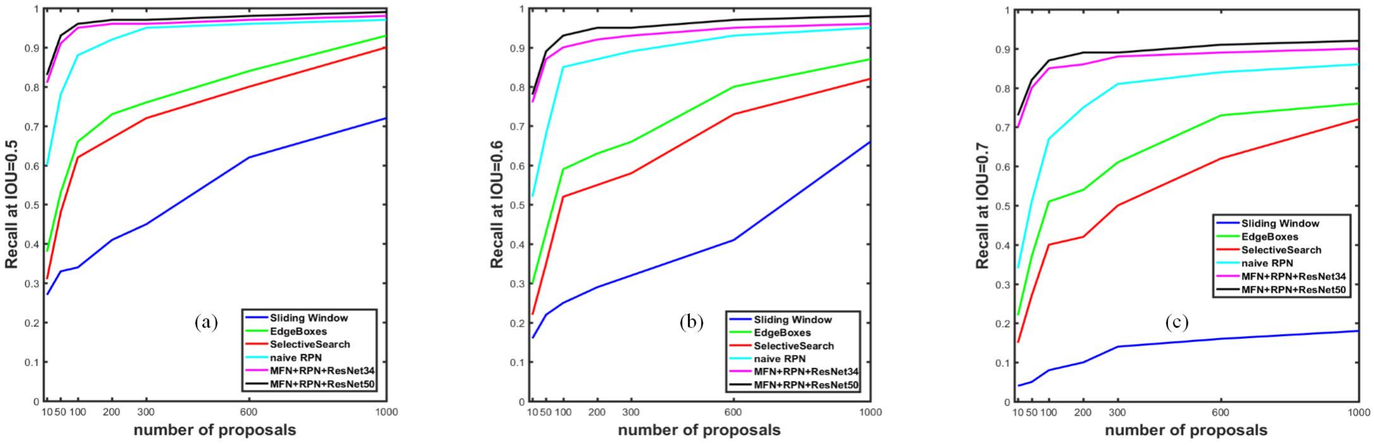
Fig. 8 shows the defect recall with various IOU thresholds at three different numbers of region proposals. The larger the IOU threshold, the more quality the selecting proposals. Unsurprisingly, the performance of the methods based on convolutional features are strongly higher than the methods without CNN [37]. When IOU > 0.7, the recall of naive RPN drops sharply compared with RPN+MFN. The naive RPN only extracts proposals from high-level features and some location information is filtered by the preceding layers making the decline of proposals in quality. With the increasing number of proposals, the naive RPN drops more sharply when IOU > 0.7. This is because RPN extract too many low-quality proposals and it is more obvious with the increase of proposals. The naive RPN works badly with the strict IOU threshold (e.g. IOU > 0.7). MFN can help RPN obtain location information from low-level and mid-level features which makes RPN is under a higher tolerance for strict IOU threshold.

Increasing the number of proposals can get a promising recall, but this will greatly increase the runtime of the detection [38]. And what is worse, low-quality proposals would be involved in the process of detection, leading to failure of defect detection in some cases. Therefore, a good detector should select as few proposals as possible and meanwhile a relatively strict IOU threshold. Fig. 9 shows the defect recalls with



(a) (b) (c)

Fig. 8. Recall vs. IOU threshold on the NEU-DET at different numbers of region proposals. (a) 50 region proposals. (b) 100 region proposals. (c) 300 region proposals.

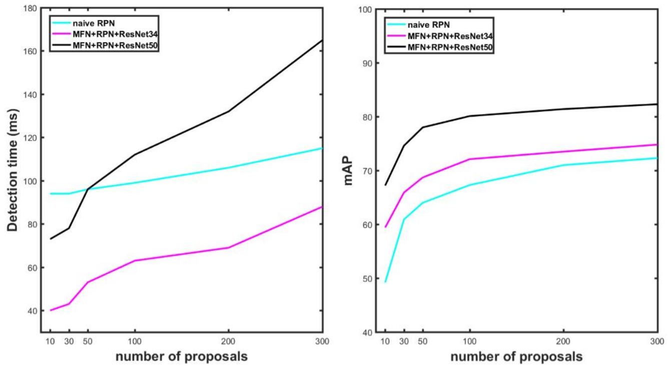


(a) (b) (c)

Fig. 9. Recall vs. number of proposals on the NEU-DET at different IOU threshold. (a) IOU threshold is 0.5. (b) IOU threshold is 0.6. (c) IOU threshold is 0.7.

various numbers of proposals at three different IOU thresholds. The naive RPN achieves a desirable recall with top-300 proposals, but RPN+MFN only need top-100 proposals to get the similar performance.

As shown in Fig. 10, for RPN+MFN with ResNet34, we achieve 92% of the performance of selecting 300 proposals by selecting only 50 proposals, which reduces the run time by half. We consider selecting top-50 proposals as a good trade-off in practical defect detection task.



(a) (b)

Fig. 10. Detection time vs. number of proposals on the NEU-DET. The detection time refers to the GPU runtime per image. Sliding window, Edge Boxes and Selective Search are CPU-based methods which are far less than GPU-based methods on detection speed.

VI. Discussion

In this section, to demonstrate our design is logical and advanced, we discuss several implicit factors which can influence on defect detection.

A. Combine which layers for MFN?

MFN combines features from various levels into a multilevel feature which is effective for improving detection. In Section III-B, it is briefly discussed that what kind of layers should be combined. In DDN, we select four layers which are the last layers of R1, R2, R3, and R4. So, whether other combination manners of these four layers may result in better performance. Therefore, we train DDN+ResNet34 in five different combination manners on NEU-DET dataset. As shown in Table III, combining all the four layers outperform the other manners. It indicates that the multilevel feature is effective for improving the accuracy of detection. Furthermore, low-level feature (e.g. R1 feature) should be paid more attention than high-level feature (e.g. R5 feature) for defect detection because R2 feature has richer location information than R5 feature.

TABLE III
Combining Layers in Different Manners

Combine layers from:				mAP:	
R2	R3	R4	R5	without L2	with L2
			√	70.2	70.2
		√	√	69.2	72.9
	√	√	√	67.0	72.9
√	√	√		59.9	73.3
√	√	√	√	58.9	74.8

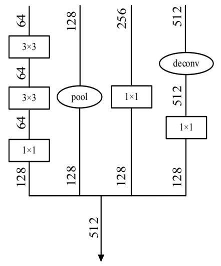
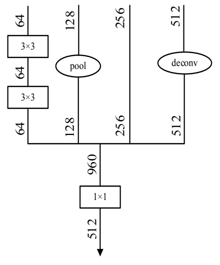
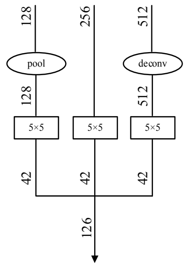
Detection mAP of DDN+ResNet34 on NEU-DET. L2: use L2 Normalization described in Section III-B.

B. Is the simple design more effective for MFN?

The major role of MFN is to uniform the features from different levels in resolution and dimensionality. To keep the dimension consistent, a straightforward approach is using 1×1 conv to reduce/increase the dimensionality. There are two placement patterns for 1×1 conv: front-mounted and back-mounted. The front-mounted pattern means 1×1 conv is placed before concatenating multilevel feature. What we use in this article is the front-mounted pattern. That is, a 1×1 conv is placed at the end of each branch of MFN. And the back-mounted pattern means a 1×1 conv is placed after concatenating multilevel feature. This pattern seems simple but in fact needs more parameters. Like [34], we use multiple 5×5 convs to uniform the resolution and dimensionality simultaneously. But the 5×5 conv is an expensive operation which has the same effect as the double stacked 3×3 conv but requiring additional parameters. Table IV shows the comparable results among three patterns in detail. The front-mounted style uses 3 times fewer parameters than the back-mounted, and 5 times fewer than Hyper-style. Therefore, MFN in the front-mounted style has less possibility to be overfitting. Moreover, in case of the same resolution size, MFN

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIM.2019.2915404, IEEE Transactions on Instrumentation and Measurement

TABLE IV
UNIFORM DIMENSIONALITY IN DIFFERENT STYLES

Front-mounted	Back-mounted	Hyper-style*
 64 3×3 64 3×3 64 1×1 128 128 pool 128 256 1×1 128 512 deconv 512 1×1 128 512	 64 3×3 64 3×3 64 128 pool 128 256 256 512 deconv 512 960 1×1 512	 128 pool 128 5×5 42 256 5×5 42 512 deconv 512 5×5 42 126
Parameters required for feature fusion network:		
1.8×10^5	5.7×10^5	9.4×10^5

All the vertical figures are the dimensionality of feature maps at the corresponding position. * Hyper-style is connected with R3, R4 and R5. But this does not mean HyperNet can work well with ResNet due to the severe dimensionality decline.

features can preserve more complete information due to its larger dimensionality than Hyper feature's (512 vs. 126).

C. Do we need more defect data?

As we known, an object detector can improve performance with more training data [39]. So, whether this rule is also effective for industrial defect data? In order to make clear this problem, we train the DDN on not only the complete NEU-DET dataset but also each subset separately. As shown in Fig. 11, for AP of each defect class, the performance of separate training is worse than the complete training in general. Specifically, the crazing, rolled-in scale, and scratches dropped sharply, whereas the inclusion, patches, and pitted surface present moderate decline. This may be due to the former requiring more data for learning than the latter. Although the total amount of training data is the same, results emerge dramatical difference. We consider that more training data can improve the represent-ability of CNN for special instances. That is to say, if DDN can be trained on more detection data, the AP may also be improved. Finally, it is need to emphasize that other types of training data maybe useless (e.g. common object) because the DDN is fine-tuned on the ImageNet pretrained model.

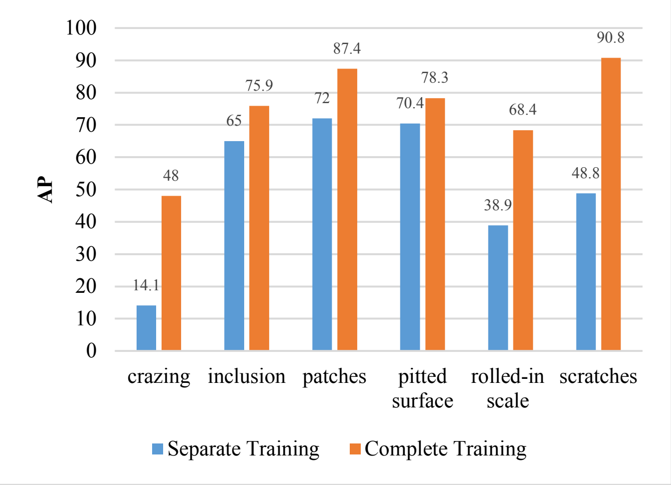
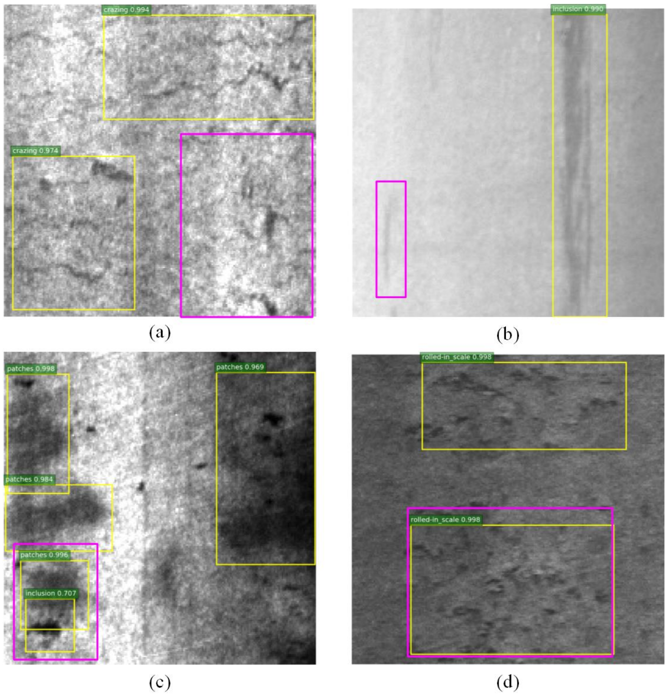


Fig. 11. The AP of each defect class on separate training vs. complete training.

D. Failure case analysis.

Though our method achieves promising results in general, in some cases there is a poor performance for defect classes like "crazing", "inclusion", "patches", and "rolled-in scale". Combining with the success cases shown in Fig. 7, we visualize some failure cases, as in Fig. 12, for analysis and attempt to explore the reasons for the unsatisfactory detection. We can observe that the DDN is robust to the continuous linear "crazing" defects but fails to find the discontinuous one in the lower right of the Fig. 12(a). It means the over-distinctive defect is hard to be correctly recognized which due to the defect data provided is not comprehensive. It is also difficult to define the confusing defects, as in Fig. 12(b), and even the human eye cannot accurately distinguish them from the background. Two kinds of defects, the "inclusion" and "patches" as in Fig. 12(c), are overlapped and the "inclusion" gets a lower score. It is no doubt that the DDN has the ability to handle the overlapped defects and the success case is shown in Fig. 7(f). We guess the reason is that the "inclusion" and the "patches" in the figure are similar, and they influence each other when they are very close. For the "rolled-in scale", the bounding-box may ignore some edge defects showed in Fig. 12(d) due to such defects are too scattered to define their scope. A more ideal defect detector is yet wanted because there is still room for improvement.



(a) (b)

(c) (d)

Fig. 12. Examples of failure cases. Yellow box indicates the detection results produced by the DDN, and pink box indicates the failure detection. (a) Over-distinctive defect; (b) Confusing defect; (c) Interference between similar defects; (d) Undefinable scope.

VII. CONCLUSION

In this paper, the defect detection network (DDN), a defect inspection system for steel plates is proposed. This system is a DL network that can obtain the specific category and detailed location of a defect by fusing the multilevel features. For defect

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIM.2019.2915404, IEEE Transactions on Instrumentation and Measurement

detection tasks, our system can provide detailed and valuable indicators for quality assessment system, such as the quantity, category, complexity, and area of a defect. Furthermore, we set up a precious defect detection dataset—NEU-DET. Experiments show that DDN can achieve 99.67% accuracy for defect classification task and 82.3 mAP for defect detection task. In addition, the system can run at a detection speed of 20 fps while keep the mAP at 70.

In the feature, we will focus on two directions as follows: The one is data augmentation technology due to the expensive manual annotations in detection datasets. The other is to perform the defect segmentation task with DL technologies, which can obtain a more precise defect boundary.

REFERENCES

[1] David Marr, *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information*. Cambridge, MA, USA: MIT, 2010, pp. 3–4.

[2] D.A. Forsyth, Computer *Vision: A Modern Approach*. Prentice Hall, 2002, pp. 482-539.

[3] K. Song and Y. Yan, "A noise robust method based on completed local binary patterns for hot-rolled steel strip surface defects," *Appl. Surface Sci.*, vol. 285, pp. 858-864, Nov. 2013.

[4] P. C. Solly, and J. E. Smith, "Adaptive surface inspection via interactive evolution," *Image Vis. Comput.*, vol. 25, no. 7, pp. 1058-1072, Jul. 2007.

[5] Y. Dong, D. Tao, and X. Li *et al*., "Texture classification and retrieval using shearlets and linear regression," *IEEE Trans. Cybern*., vol.45, no.3, pp. 359-369, Mar. 2015.

[6] M. Xiao, M. Jiang, and G. Li *et al*., "An evolutionary classifier for steel surface defects with small sample set," *EURASIP J. Image Vid. Process.*, vol. 2017, no. 1, pp. 48-61, Dec. 2017.

[7] Y. Park, and I.S. Kweon, "Ambiguous surface defect image classification of AMOLED displays in smartphones," *IEEE Trans. Ind. Informat*., vol. 12, no. 2, pp. 597-607, Apr. 2016.

[8] M. Chu, J. Zhao, and X. Liu *et al*., "Multi-class classification for steel surface defects based on machine learning with quantile hyper-spheres," *Chemom. Intell. Lab. Syst*., vol. 168, pp. 15-27, Sep. 2017.

[9] S. Ghorai, A. Mukherjee, and M. Gangadaran *et al*., "Automatic defect detection on hot-rolled flat steel products," *IEEE Trans. Instrum. Meas.*, vol. 62, no. 3, pp. 612-621, Mar. 2013.

[10] Q. Luo and Y. He, "A cost-effective and automatic surface defect inspection system for hot-rolled flat steel," *Robot. Comput.-Integr. Manuf.*, vol. 38, pp. 16–30, Apr. 2016.

[11] K. Liu, H. Wang, and H. Chen, *et al*. "Steel surface defect detection using a new haar weibull-variance model in unsupervised manner. *IEEE Trans. Instrum. Meas.*, vol. 66, no. 10, pp. 2585-2596, Oct. 2017.

[12] M. Chu, R. Gong, S. Gao *et al*., "Steel surface defects recognition based on multi-type statistical features and enhanced twin support vector machine," *Chemom. Intell. Lab. Syst*., vol.171, pp. 140-150, Sep. 2017.

[13] He K , Gkioxari G , and Dollar P , *et al*., "Mask R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Venice, Italy, Oct. 2017, pp. 2980-2988.

[14] S. Ren, K. He, and R. Girshick *et al*., "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Proc. Neural Inf. Process. Syst. (NIPS)*, QA, CAN, Dec. 2015, pp. 91-99.

[15] Yosinski J , Clune J , and Bengio Y , *et al*., "How transferable are features in deep neural networks?," in *Proc. Neural Inf. Process. Syst. (NIPS)*, Montréal, CAN, Dec. 2014, pp. 3320-3328.

[16] Y. Lecun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436-444, May 2015.

[17] R. Ren, T. Hung, and K.C. Tan, "A generic deep-learning-based approach for automated surface inspection," *IEEE Trans. Cybern.*, vol. 48, no. 3, pp. 929-940, Mar. 2018.

[18] Y. Li, G. Li, and M. Jiang, "An end-to-end steel strip surface defects recognition system based on convolutional neural networks," *Steel Research Int.*, vol. 88, no.2, pp. 176-187, Feb. 2017.

[19] S. Zhou, Y. Chen, and D. Zhang, "Classification of surface defects on steel sheet using convolutional neural networks," *Materiali in Tehnologije,* vol. 51, no.1, pp. 123-131, Feb. 2017.

[20] V. Natarajan, T.Y. Hung, and S. Vaikundam *et al*., "Convolutional networks for voting-based anomaly classification in metal surface inspection," in *Proc. IEEE Int. Conf. Ind. Technol. (ICIT)*, ON, CAN, Mar. 2017, pp. 986-991.

[21] P. Chen, and S.S. Ho, "Is overfeat useful for image-based surface defect classification tasks?" in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, AZ, USA, Sep. 2016, pp. 749-753.

[22] J. Deng, W. Dong, and R. Socher *et al*., "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Comput. Vis. Pattern Recognit. (CVPR)*, Anchorage, AK, Jun. 2009, pp. 248-255.

[23] K. He, X. Zhang, and S. Ren, "Deep residual learning for image recognition," in *Proc. IEEE Comput. Vis. Pattern Recognit. (CVPR)*, Boston, MA, USA, Jun. 2015, pp. 770-778.

[24] P. Sermanet, D Eigen, and X Zhang *et al*., "OverFeat: Integrated recognition, localization and detection using convolutional networks," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, Banff, AB, Canada, Apr. 2014, pp. 1-16.

[25] A. Krizhevsky, I. Sutskever, and G.E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Neural Inf. Process. Syst. (NIPS)*, vol. 60, no. 2, NV, USA, Dec. 2012, pp. 1097-1105.

[26] C. Szegedy, V. Vanhoucke, and S. Ioffe *et al*., "Rethinking the Inception architecture for computer vision," in *Proc. IEEE Comput. Vis. Pattern Recognit. (CVPR)*, NV, USA, Jun. 2016, pp. 2818-2826.

[27] R. Girshick, J. Donahue, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proc. IEEE Comput. Vis. Pattern Recognit. (CVPR)*, Columbus, OH, USA, Jun. 2014, pp. 580-587.

[28] R. Girshick, "Fast R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Santiago, Chile, Dec. 2015, pp. 1440-1448.

[29] J. Redmon, S. Divvala, and R. Girshick *et al*., "You only look once: Unified, real-Time object detection," in *Proc. IEEE Comput. Vis. Pattern Recognit. (CVPR)*, NV, USA, Jun. 2016, pp. 779-788.

[30] W. Liu, D. Anguelov, and D. Erhan *et al*., "SSD: Single shot multibox detector," in *Proc. Springer Euro. Conf. Comput. Vis. (ECCV)*, Amsterdam, the Netherlands, Oct. 2016, pp. 21-37.

[31] L. Zhang, Y. Gao, C. Hong *et al*., "Feature correlation hypergraph: exploiting high-order potentials for multimodal recognition," *IEEE Trans. Cybern.*, vol. 44, no. 8, pp. 1408-1419, Aug. 2014.

[32] Y. Lecun, L. Bottou, and Y. Bengio *et al*., "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278-2324, Nov. 1998.

[33] X. Glorot, and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," *J. Mach. Learn. Res.*, vol. 9, pp. 249-256, 2010.

[34] T. Kong, A. Yao, Y. Chen, and F. Sun, "HyperNet: Towards accurate region proposal generation and joint object detection." in *Proc. IEEE Comput. Vis. Pattern Recognit. (CVPR)*, NV, USA, Jun. 2016, pp. 845-853.

[35] C.L. Zitnick, and P. Dollar, "Edge boxes: Locating object proposals from edges," in *Proc. Springer Euro. Conf. Comput. Vis. (ECCV)*, Zurich, Switzerland, Oct. 2014, pp. 391-405.

[36] J.R.R. Uijlings, K.E.A.V.D. Sande, and T. Gevers *et al.*, "Selective search for object recognition," *Int. J. Comput. Vis.*, vol. 104, no. 2, pp. 154-171, Sep. 2013.

[37] Y. Wei, Y. Zhao, and C. Lu *et al*., "Cross-modal retrieval with CNN visual features: A New Baseline," *IEEE Trans. Cybern.*, vol. 47, no. 2, pp. 449-460, Feb. 2017.

[38] J. Hosang, R. Benenson, and P. Dollar *et al*., "What makes for effective detection proposals?" *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 4, pp. 814-830, Apr. 2016.

[39] X. Zhu, C. Vondrick, and C.C. Fowlkes et al., "Do we need more training data?" *Int. J. Comput. Vis.*, vol. 119, no. 1, pp. 76-92, 2016.

[40] K. Simonyan, and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, CA, USA, May 2015, pp. 1-16.

Yu He received the B.S. degree in School of Mechanical Engineering and Automation, Liaoning Technical University, Fuxin, China, in 2014, and the M.S. degree in School of Mechanical Engineering and Automation, Northeastern University, Shenyang, China, in 2016. He is currently pursuing the Ph.D. degree with School of Mechanical Engineering and Automation, Northeastern University, China. His research interests include deep learning, pattern recognition and intelligent inspection.

Kechen Song received the B.S., M.S. and Ph.D. degrees in School of Mechanical Engineering and Automation, Northeastern University, Shenyang, China, in 2009, 2011 and 2014, respectively. He has been a teacher in Northeastern University of China since 2014. His research interest covers vision-based inspection system for steel surface defects, surface topography, image processing and pattern recognition.

Qinggang Meng received the B.S. and M.S.degrees from the School of Electronic Information Engineering, Tianjin University, China, and the Ph.D. degree in computer science from Aberystwyth University, U.K. He is a Reader with the Department of Computer Science, Loughborough University, U.K. His research interests include biologically and psychologically inspired learning algorithms and developmental robotics, service robotics, robot learning and adaptation, multi-UAV cooperation, drivers distraction detection, human motion analysis and activity recognition, activity pattern detection, pattern recognition, artificial intelligence, and computer vision. He is a fellow of the Higher Education Academy, U.K.

Yunhui Yan received the B.S., M.S. and Ph.D. degrees in School of Mechanical Engineering and Automation, Northeastern University, Shenyang, China, in 1981, 1985 and 1997, respectively. He has been a teacher in Northeastern University of China since 1982, and became as professor in 1997. During 1993-1994, he stayed in the Tohoku National Industrial Research Institute as a visiting scholar. His research interest covers intelligent inspection, image processing and pattern recognition.