Results in Engineering 25 (2025) 103972

Contents lists available at ScienceDirect

Results in Engineering

journal homepage: www.sciencedirect.com/journal/results-in-engineering

Research paper

Steel surface defect detection and segmentation using deep neural networks

Sara Ashrafi [a], Sobhan Teymouri [b], Sepideh Etaati [b], Javad Khoramdel [a], Yasamin Borhani [a], Esmaeil Najafi [c,*]

[a] Center of Excellence in Robotics and Control, Advanced Robotics and Automated Systems (ARAS), Faculty of Mechanical Engineering, K. N. Toosi University of Technology, Tehran, Iran
[b] Faculty of Electrical Engineering, K. N. Toosi University of Technology, Tehran, Iran
[c] Smart Mechatronics and Robotics Research Group, Saxion University of Applied Sciences, Enschede, the Netherlands

ARTICLE INFO

Keywords:
Steel surface defect detection
Computer vision
Semantic segmentation
Object detection

ABSTRACT

Defect detection is a crucial task in the manufacturing industry, particularly in steel surface inspection. While manual recognition is one of the most reliable techniques, recent advances in computer vision and machine learning have led to the development of automatic defect detection techniques. This paper proposes several deep-learning-based computer vision techniques, including semantic segmentation and object detection models, to detect surface defects on steel sheets. The U-Net, FCN-8, and FPN models are implemented for segmentation, while the YOLOv4 model is used for object detection. Moreover, a combined segmentation and object detection structure, referred to as two-stage defect detection, is developed to enhance the accuracy of detecting small defects. Based on the obtained results, the U-Net model with pre-trained backbones achieves a Dice Similarity Coefficient of 72%, outperforming existing methods. The object detection model with a resolution of 640 reaches the mean average precision of 49.32% and 35.06% for binary class and multi-class detection, respectively. Furthermore, the proposed two-stage defect detection structure achieves a Dice Similarity Coefficient of 84%. In summary, the results validate the efficient performance of the studied techniques for accurate defect detection on steel surfaces.

1. Introduction

Steel plays an important role as a strategic material in the industry, used in manufacturing from cars and ships to buildings and bridges. In general, steel is not protected from defects and damages that can decrease its structural integrity and performance. These defects can occur during the manufacturing process, transportation, storage, and usage while taking different forms such as cracks, scratches, dents, and corrosion.

Early detection and localization of these defects is crucial for ensuring the safety of steel products, reliability, and quality. This task has been performed using mechanical and chemical methods, such as ultrasonic testing [1], magnetic particle inspection [2], and dye penetrant testing [3], which rely on physical and chemical properties of the defects. However, these methods can be time-consuming, and costly, and may not always detect small surface defects.

In recent years, there has been growing interest in using computer vision techniques, specifically neural networks, to detect and segment steel defects from images of the steel surface [4]. Using neural networks leverages advances in image processing, machine learning, and deep-learning to automate and improve defect detection. Unlike traditional methods, image-based defect detection can be non-destructive, faster, and more accurate, capturing surface and subsurface defects.

In this paper, different methods of image processing, like segmentation, and object detection, are used for detecting steel surface defects. For the segmentation task, U-Net [5], Fully Convolutional Network-8 (FCN-8) [6], and Feature Pyramid Network (FPN) [7] are implemented on the SVERESTAL steel defect dataset [8]. The models should be able to detect and classify the defects of the steel. Moreover, a method is proposed to solve the difficult problem of detecting small or subtle defects: a two-stage detection and segmentation model. First, defects are detected by the YOLOv4 [9] object detection model, and then the cropped images will be segmented using the U-Net architecture. The proposed two-stage defect detection structure reaches a Dice Similarity Coefficient of 84%, which outperforms the defect accuracy specifically for the small-size defects.

* Corresponding author.
E-mail address: e.najafi@saxion.nl (E. Najafi).

https://doi.org/10.1016/j.rineng.2025.103972
Received 4 August 2024; Received in revised form 10 December 2024; Accepted 3 January 2025

Available online 9 January 2025

The structure of the rest of this paper is as follows: Section 2 will review related works, Section 3 will illustrate the training datasets and their preparation for training segmentation and object detection models, Section 4 will present the semantic segmentation models, and object detection model used to train the dataset, and will describe a two-stage method that combines object detection and segmentation, Section 5 will discuss the results and discussion. Finally, Section 6 will conclude the paper with a summary of the main findings and implications.

2. Related works

The detection of defects in steel has been widely studied, with researchers exploring various methods, including signal processing, computer vision, and deep-learning. Early research primarily focused on signal-processing techniques. In [10], several signal processing methods for pulsed thermography were compared, including principal component thermography (PCT), partial least squares thermography (PLST), and time-frequency analysis (TFA).

Recently, computer vision and image processing techniques have gained attention for steel defect detection. In [11], a method for classifying surface defects in steel using an ensemble approach based on deep convolutional neural networks (CNNs) [12] was proposed. Similarly, [13] employed CNNs and vision transformers [14] to classify steel defects. However, defect classification alone is insufficient for practical purposes, as the precise location of the defect is crucial to determine the extent of damage and to implement necessary corrective actions.

Semantic segmentation has been utilized to achieve precise defect localization. It involves partitioning an image into multiple segments based on semantics to identify regions of interest. One of the practical methods for semantic segmentation problems is U-Net [5], a CNN architecture originally designed for biomedical image segmentation. The U-Net architecture has been widely used for various image segmentation tasks and has shown promising results in steel defect detection.

Several studies have applied U-Net architectures for detecting steel defects. In [15], U-Net-like architectures with various encoders were evaluated on the SEVERSTAL Steel dataset. Using the Dice Similarity Coefficient (DSC) and Intersection over Union (IoU) metrics, the models achieved a DSC of 0.9304 and an IoU of 0.9122 with the U-Net model incorporating a ResNet152 [16] backbone. However, the training dataset in this study was a combination of two datasets, and only Pitted surface and Scratch defects (classes 1 and 3) were used, making direct comparison challenging. Another study [17] investigated the impact of surface lighting levels on defect detection performance using a U-Net-based neural network with a ResNet152 decoder. The best recognition result on the SEVERSTAL dataset was achieved at an illumination level of 300 lx, with a DSC of 0.89. Additionally, U-Net architectures with ResNet and DenseNet backbones were used in [18], with models trained on the SEVERSTAL dataset.

In [19], the authors aimed to improve steel defect segmentation using deep-learning with U-Net and FPN architectures. Both architectures were evaluated with different backbone encoders. For U-Net, the authors reported that the ResNet34 backbone achieved a DSC of 0.554 and an IoU of 0.389, while the EfficientNet backbone performed better, achieving a DSC of 0.709 and an IoU of 0.554. Similarly, for FPN, the ResNet34 backbone achieved a DSC of 0.521 and an IoU of 0.425, while EfficientNet again outperformed ResNet34, achieving a DSC of 0.698 and an IoU of 0.535. These findings highlight the importance of selecting an appropriate backbone encoder to enhance segmentation accuracy in defect detection tasks.

Other methods such as DeepLab [20] and Generative Adversarial Networks (GANs) [21] have also been utilized for surface defect detection. In [22], the performance of DeepLabV3+ with various backbones on the SEVERSTAL dataset was analyzed. Employed with random weighted augmentation to balance different types of defects in the training set, the study tested backbones including ResNet, DenseNet, and EfficientNet. ResNet and EfficientNet yielded the best IoU scores on the test set, around 0.57. Another study [23] proposed a semi-supervised deep-learning model named SSGAN for pixel-wise segmentation of steel surface defects. The model, based on a GAN trained with both labeled and unlabeled data, achieved mean IoUs of 79.0% and 81.8% using 1/8 and 1/4 labeled data, respectively, on the SEVERSTAL dataset.

Object detection models have also been employed to identify defects on steel surfaces. A recent approach [24] utilized an improved YOLOv4 neural network to detect surface defects on steel rails. Training the network on images of steel rails with surface defects, the method achieved an accuracy rate of 92.68%. However, detecting defects and drawing bounding boxes may not always produce precise outcomes, as the bounding box may encompass non-defective areas. In such cases, combining object detection with segmentation techniques can enhance precision.

To address these limitations, some studies have proposed deep-learning-based techniques that integrate object detection and segmentation. One such study introduced the Forceful Steel Defect Detector (FDD) system [4], aiming to enhance defect detection accuracy by incorporating advanced techniques such as deformable convolution, guided anchoring region proposals, and scaling techniques. The FDD system was evaluated on three datasets—the SEVERSTAL steel dataset, the NEU steel dataset [25], and the DAGM dataset [26]—achieving high detection rates with Average Precision (AP) scores of 96.9%, 99.6%, and 100%, respectively.

Training schemes for two-stage neural networks have also been explored for defect detection [27,28]. The end-to-end simultaneous learning method proposed in [27] balances segmentation and classification losses throughout the learning process, utilizing the AP metric for evaluation. The network was trained for binary classification—defective (positive) and non-defective (negative) data—and applied to the DAGM dataset, the KolektorSDD dataset [29], and the SEVERSTAL dataset. It achieved detection rates of 100% on the DAGM and KolektorSDD datasets, and 98.74% on the SEVERSTAL dataset using the AP metric. For the SEVERSTAL dataset, only one class of defects (type 3) was used, as it contains the highest number and most common defects. In [28], a two-stage method focusing primarily on object detection was applied to wheat disease detection.

Surface defect detection techniques have also been applied to other industries. In [30], an automatic surface defect inspection system for automobiles using computer vision methods was presented. The system employed a camera to capture images of car body surfaces and utilized image processing techniques to detect defects such as scratches, dents, and corrosion, achieving an accuracy rate of 92.3%.

In this paper, three different segmentation architectures are implemented—U-Net, FCN-8, and FPN—in combination with the object detection model YOLOv4, applying them in a two-stage process of object detection followed by segmentation. Specifically, a method integrating YOLOv4 and U-Net is employed and evaluated on the SEVERSTAL dataset to assess its effectiveness.

3. Dataset and features

The SEVERSTAL Steel Defect Detection dataset is a publicly available dataset designed for defect detection in steel surfaces. This dataset was introduced as part of a Kaggle competition organized by the steel company SEVERSTAL. It consists of high-resolution images of steel surfaces with various types of defects, providing a challenging environment for deep-learning models to detect defects in steel surfaces. Fig. 1 depicts a random sample of the dataset showcasing defects from the first defect class.

The dataset is divided into two main subsets: a training set and a test set. The training set contains approximately 12,000 images, while the test set contains approximately 1,800 images. Each image has a size of 1600×256 pixels. According to [4], the dataset includes four classes of defects:

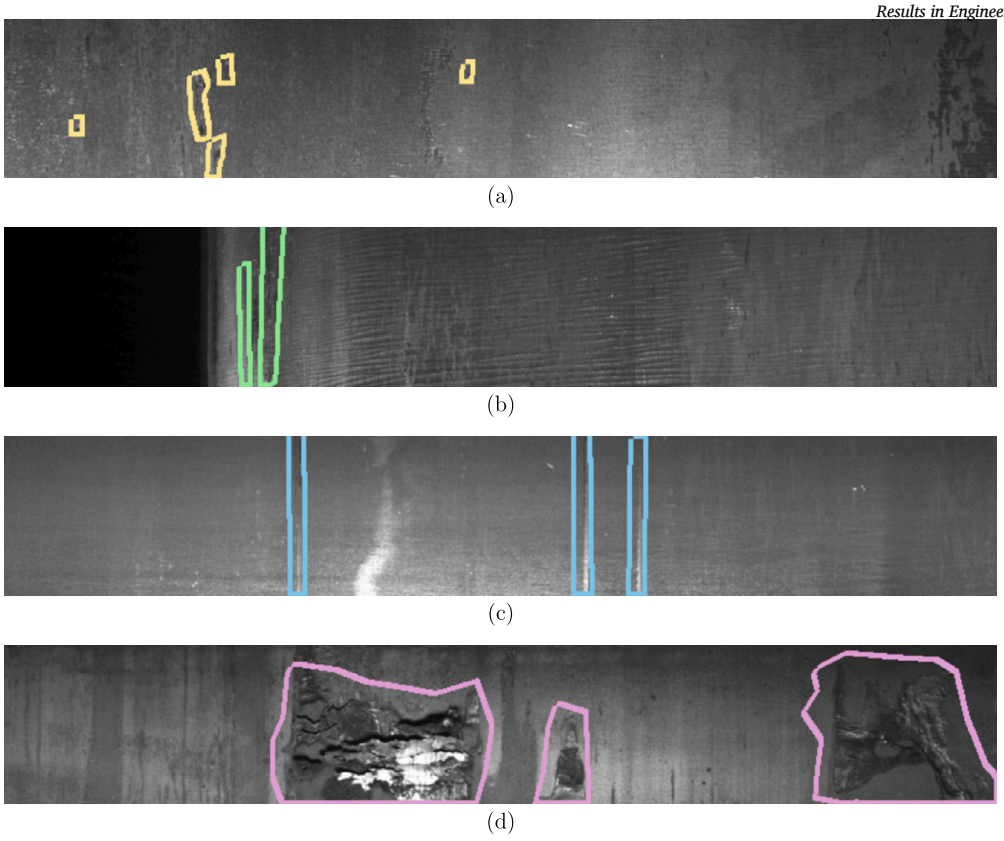

Fig. 1. Sample Images from SEVERSTAL Steel Defect Detection Dataset. (a) Defect from class 1 (Pitted surface) marked with yellow, (b) Defect from class 2 (Crazing) marked with green, (c) Defect from class 3 (Scratch) marked with blue, (d) Defect from class 4 (Patch) marked with purple.

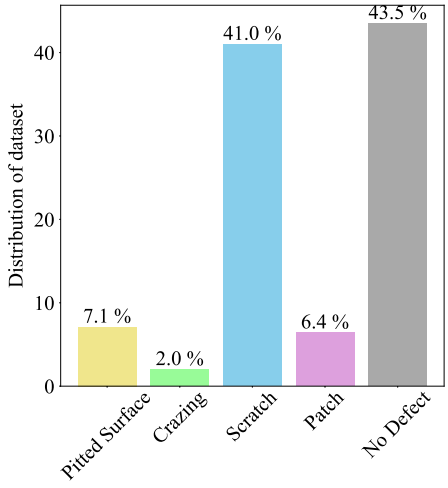

Fig. 2. Distribution of defect classes in SEVERSTAL Steel Defect Detection Dataset. The bar chart illustrates the distribution of different defect classes in the SEVERSTAL Steel Defect Detection dataset.

- **Class 1**: Pitted surface defects
- **Class 2**: Crazing defects
- **Class 3**: Scratch defects
- **Class 4**: Patch defects

Each image can have multiple defects, and the labels indicate the presence of each defect type. However, the dataset has an imbalanced class distribution, which poses challenges during model training.

Fig. 2 illustrates the distribution of classes within the dataset. The results indicate that the third defect class accounts for 41% of the entire dataset, while the second defect class accounts for just 2%. These findings suggest that the third and second defect classes are the most and least represented classes, respectively. Additionally, a significant proportion of the dataset (43.5%) consists of images without any defects.

The imbalanced defect distribution significantly impacts the training process. Since **Class 3** has a substantially higher number of samples, the model tends to learn and predict **Class 3** defects more effectively. In contrast, classes with fewer samples, such as **Class 2**, may be underrepresented during training, leading to poorer performance in detecting these defects. Moreover, the size of the defects varies among classes; **Class 1** defects are the smallest, making them more challenging to detect compared to larger defects in other classes. This variation in defect size adds complexity to the training process, as the model must learn to identify defects of different scales.

By acknowledging and addressing the challenges posed by the imbalanced class distribution and varying defect sizes, this study aims to develop a more robust model that performs well across all defect types in the SEVERSTAL Steel Defect Detection dataset.

4. Intelligent defect detection and segmentation models

In this section, three different segmentation models are evaluated on the SEVERSTAL dataset. Moreover, a method for achieving more accurate segmentation results is introduced. The first step involves the implementation of an object detection model on the dataset. Subsequently, a two-stage model is proposed to segment the defective areas.

4.1. Semantic segmentation models

Semantic segmentation is a computer vision technique that involves dividing an image into different regions or segments and labeling each segment with its corresponding object category. The goal of semantic segmentation is to accurately identify the objects present in the image

and their boundaries. In this paper, three different segmentation methods are applied to the SEVERSTAL dataset: U-Net, Fully Convolutional Network-8 (FCN-8), and Feature Pyramid Network (FPN).

4.1.1. U-Net

The U-Net model is a type of convolutional neural network (CNN) architecture that was introduced in [5] for biomedical image segmentation. It is called U-Net because of its U-shaped architecture, which consists of an encoder and a decoder. The encoder part of the network extracts high-level features from the input image, while the decoder part up-samples the features to the original image size and generates the segmentation map.

The U-Net architecture is commonly used for image segmentation tasks. In this paper, pre-trained backbones such as EfficientNet-B0 and ResNet34 are utilized for the U-Net architecture. Transfer learning is employed by using pre-trained weights from the ImageNet dataset, which significantly reduces the amount of training required. By removing the last few layers responsible for classification and appending the U-Net architecture to the backbone, the pre-trained weights can be leveraged to initialize the U-Net weights. This results in a highly accurate model for image segmentation tasks even with relatively few training examples.

4.1.2. Fully convolutional network-8

Fully Convolutional Network-8 (FCN-8) is another type of CNN architecture introduced in [6] for semantic segmentation. It is a fully convolutional version of the popular VGG-16 architecture, where the fully connected layers are replaced with convolutional layers. The FCN-8 architecture uses a combination of skip connections and upsampling to generate the final segmentation map. In this paper, FCN-8 with a pre-trained VGG16 backbone is trained on the SEVERSTAL dataset.

4.1.3. Feature pyramid network

Feature Pyramid Network (FPN) is an architecture introduced in [7], which enables the network to aggregate features from different scales. FPN is designed to handle objects at various scales by creating a feature pyramid where features from different layers of the network are combined to generate a multi-scale representation of the input image. The final segmentation map is generated by applying convolutional layers to the feature pyramid.

In this paper, FPN is used with ResNet34 and EfficientNet-B0 as backbones for segmentation. FPN uses a top-down architecture to combine features from different levels of a CNN into a single feature pyramid. ResNet34 is deeper and more complex, while EfficientNet-B0 is more efficient and requires fewer parameters.

4.1.4. Pretraining and training configuration

For all segmentation models, the input images are resized to a resolution of 128×800 pixels, which is half the original resolution, to reduce computational cost during training. The dataset is divided into 80% for training and 20% for validation. The batch size is set to 32 for training the models. The models are trained using the Adam optimizer, with an initial learning rate of 10^{-3}, which is reduced exponentially during training until it reaches 10^{-8}.

Transfer learning is employed by using pre-trained weights from the ImageNet dataset, which significantly reduces the amount of training required. By removing the last few layers responsible for classification and appending the segmentation architecture to the backbone, the pre-trained weights can be leveraged to initialize the model weights.

To prevent overfitting, early stopping and model checkpoint callbacks were employed during training. The model's performance on the validation set was monitored, and the model with the minimum validation loss was saved. Although the models were set to train for up to 30 epochs, the best models were often achieved at earlier epochs, as indicated by the validation metrics. Due to early stopping and convergence:

1. The **U-Net with EfficientNet-B0 and ResNet34 backbones** trained for 15 epochs and 20 epochs, respectively.
2. The **FCN-8 and FCN-8 with VGG16 backbone** trained for 24 epochs and 19 epochs, respectively.
3. The **FPN with EfficientNet-B0 and ResNet34 backbones** trained for 17 epochs and 23 epochs, respectively.

This approach ensures that the final models used for evaluation are the ones with the best generalization performance, mitigating the effects of overfitting.

Fig. 3 shows the training and validation loss curves, as well as the DSC and IoU metrics over the epochs for each of the respective models with the best results. Including these graphs in this section helps justify the selection of training parameters and illustrates the point at which overfitting begins to occur.

These figures demonstrate how monitoring the validation loss allows detection of the onset of overfitting and enables taking corrective action through early stopping, ensuring that the models generalize well to unseen data.

The loss function used for training is a combination of Binary Cross Entropy (BCE) and Dice Loss (DL), as defined in Equation (5). The evaluation metrics are the DSC (DSC) and Intersection over Union (IoU), as described in Subsection 4.1.5.

4.1.5. Evaluation metrics and loss function

To evaluate the performance of steel defect segmentation, two metrics are used: Intersection over Union (IoU) [31] and Dice Similarity Coefficient (DSC) [32]. The IoU coefficient is a measure of the overlap between two segmentation masks. It is calculated as the intersection of the ground-truth mask and the predicted mask, divided by the union of the two masks. The IoU score is calculated for each of the four classes using

$$IoU = \frac{|X \cap Y|}{|X \cup Y|} = \frac{|X \cap Y|}{|X| + |Y| - |X \cap Y|} \tag{1}$$

where X is the ground truth mask, Y is the predicted segmentation mask, and $IoU \in [0, 1]$.

The DSC metric is a measure of the similarity between two segmentation masks and is defined by

$$DSC = \frac{2|X \cap Y|}{|X| + |Y|} \tag{2}$$

with $DSC \in [0, 1]$.

The Dice Loss (DL) [33] is a commonly used loss function in semantic segmentation tasks to address the issue of class imbalance in the dataset. The DL penalizes the difference between the predicted segmentation mask and the ground truth mask and is derived from the DSC as follows:

$$DL = 1 - DSC = 1 - \frac{2|X \cap Y|}{|X| + |Y|} \tag{3}$$

Binary Cross Entropy (BCE) is another loss function commonly used in semantic segmentation, especially for binary classification tasks [34]. BCE measures the discrepancy between the predicted probabilities and the actual binary labels and is defined by

$$BCE = -(X \log(Y) + (1 - X) \log(1 - Y)) \tag{4}$$

where $X \in \{0, 1\}$ is the ground truth label, and $Y \in [0, 1]$ is the predicted probability. BCE effectively penalizes confident but incorrect predictions, aiding the model in learning from misclassifications.

The loss function employed for model training involves combining Dice Loss and Binary Cross Entropy through a weighted sum, resulting in the overall loss function. The intent behind this approach is to leverage the ability of Dice Loss to handle class imbalances and the capacity of BCE for producing smoother gradients [35]. The equation for the combined loss function is presented by

$$\text{Loss} = BCE + DL \tag{5}$$

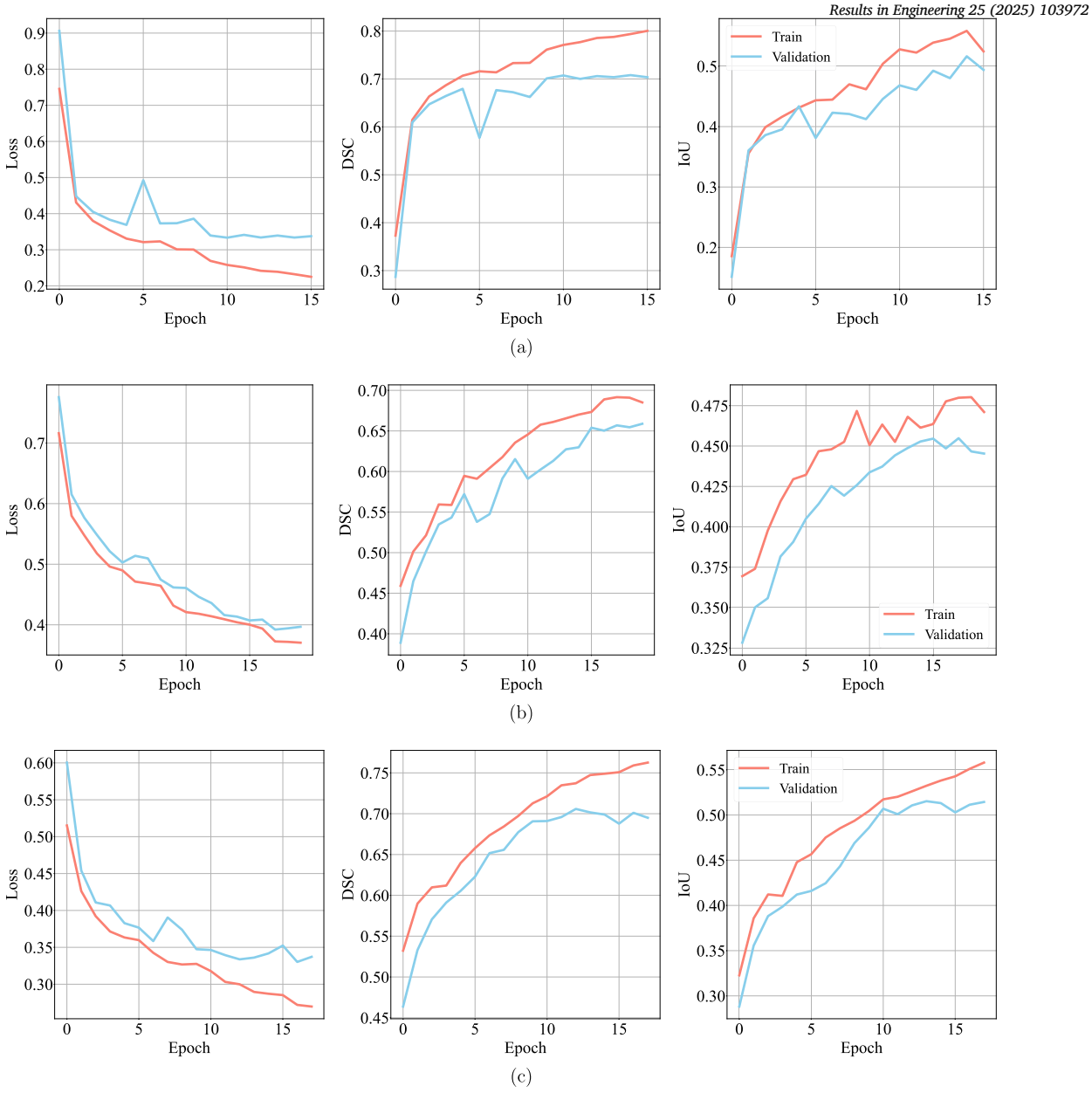

Fig. 3. Training metrics of the Semantic Segmentation models. (a) U-Net model with EfficientNet-B0 backbone, (b) FCN-8 model with VGG16 backbone, (c) FPN model with EfficientNet-B0 backbone. From left to right: BCE Dice loss, DSC, and IoU curves during the training process.

4.2. Object detection model

Object detection is a computer vision task that involves identifying the presence and location of objects in an image or video. Unlike semantic segmentation, which labels each pixel in the image with its corresponding object category, object detection provides a bounding box around each object along with its class label.

YOLOv4 is an efficient object detection algorithm introduced in [9]. YOLO stands for "You Only Look Once," meaning the network takes the entire image as input and predicts the bounding boxes and class probabilities for all objects in a single forward pass. YOLOv4 uses a deep neural network architecture composed of multiple convolutional layers and employs various techniques to improve accuracy, including anchor boxes, Feature Pyramid Networks (FPN), and the Mish activation function. In this paper, the YOLOv4 algorithm is used to detect defects on steel sheets.

To detect defects, two different object detection approaches are used:

1. **Binary Class Detection**: An algorithm that detects defects of any kind without differentiating between defect types. This simplifies the problem to identifying whether a defect is present or not.
2. **Multi-Class Detection**: An algorithm that detects and classifies four types of defects corresponding to the classes in the dataset. This approach provides more detailed information about the defect type.

4.2.1. Pretraining and training configuration

The YOLOv4 models are trained with input resolutions of 416×416, 544×544, 608×608, and 640×640 pixels. The models are trained from scratch because using pre-trained weights led to divergence during training. The batch size is set to 8 for the 416×416 resolution and 4 for the higher resolutions due to memory constraints.

The Adam optimizer is employed with an initial learning rate of 10^{-4}, which is gradually reduced to 10^{-6} during training. To prevent overfitting, early stopping and model checkpoint callbacks are utilized. The model's performance on the validation set is monitored, and the model

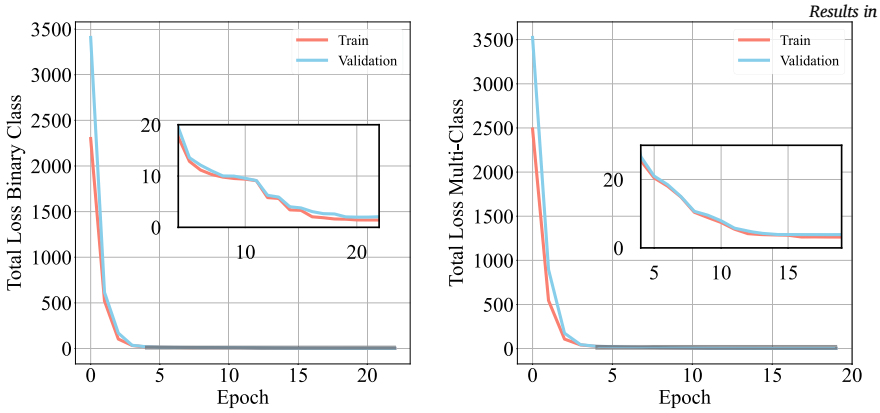

Fig. 4. Total Loss Curves During YOLOv4. The total loss change curve during the training process of the YOLOv4 model with a resolution of 640 for binary class (left) and multi-class (right) approaches.

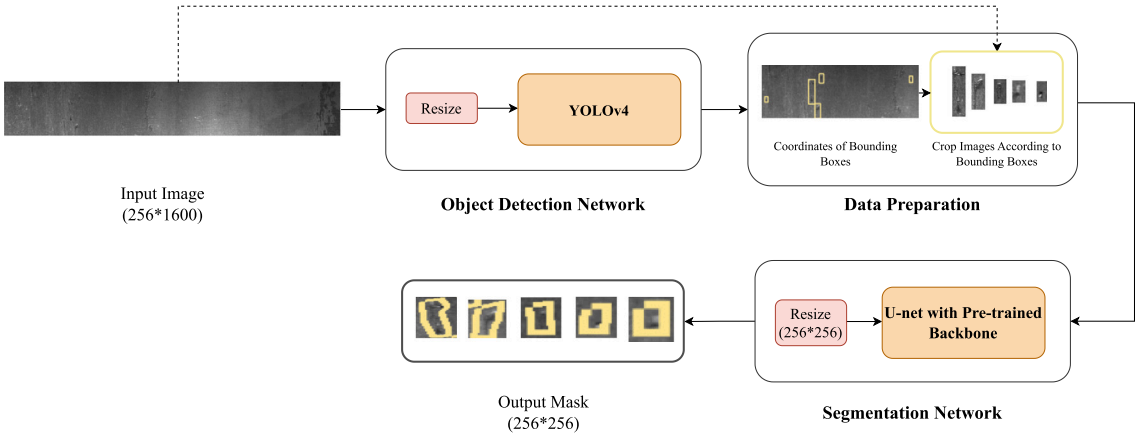

Fig. 5. Structure of Two-Stage Defect Detection and Segmentation Model. The first stage uses YOLOv4 for object detection to extract bounding box coordinates of defected areas. The original image is cropped with a 10-pixel margin around each bounding box. The second stage uses a U-Net with a pre-trained EfficientNet-B0 backbone to segment the defected areas from the cropped images.

with the lowest validation loss is saved. Although the models were set to train for up to 30 epochs, due to early stopping and heavy computational demands, the binary class model effectively trained for 26 epochs, and the multi-class model for 19 epochs. This ensures that the final models have optimal generalization performance and reduces the risk of overfitting.

Data augmentation techniques, including horizontal flipping, cropping, and translation, are applied during training to enhance model robustness. The performance of the object detection models is evaluated using the Mean Average Precision (mAP) metric.

Fig. 4 includes the training and validation loss curves for the YOLOv4 model for binary and multi-class detection with a resolution of 640, demonstrating how the model's performance was monitored to prevent overfitting.

4.2.2. Evaluation metrics

Average Precision (AP) is a metric used to evaluate the performance of object detection models for a single class. It summarizes the precision-recall curve as the weighted mean of precisions achieved at different recall levels [36]. AP is calculated using the following equation:

$$AP = \int_0^1 p(r)\,dr \tag{6}$$

where $p(r)$ is the precision as a function of recall r.

Mean Average Precision (mAP) is the mean of the Average Precision values across all classes, providing a single metric to evaluate the overall performance of the object detection model. It is defined as:

$$mAP = \frac{1}{N}\sum_{i=1}^{N} AP_i \tag{7}$$

where N is the number of classes, and AP_i is the Average Precision for class i. In this work, mAP is calculated to assess how well the object detection model performs across all defect types.

4.3. Two-stage defect detection structure

In addition to the segmentation and object detection networks, a combination of object detection and segmentation tasks is being investigated. In this proposed method, a two-stage defect detection model has been developed using YOLOv4 and U-Net with a pre-trained backbone.

As it is depicted in Fig. 5, in the first stage of the model, an object detection algorithm is applied to the original input image. The algorithm extracts the corresponding coordinates of the bounding boxes that identify the defected areas in the image. The original image is then cropped with a 10-pixel margin around each bounding box using these coordinates. In the second stage, a segmentation algorithm is implemented on the cropped images to accurately detect and segment the defected area. The goal of this approach is to enable the model to identify and isolate the defected area more precisely and with greater accuracy than using either object detection or segmentation alone.

4.3.1. Pretraining and training configuration

To apply this two-stage model to the dataset, the original images are first cropped according to the bounding boxes obtained from the object detection algorithm. The cropped images are then fed into a U-Net with an EfficientNet-B0 backbone. In the training process, all images

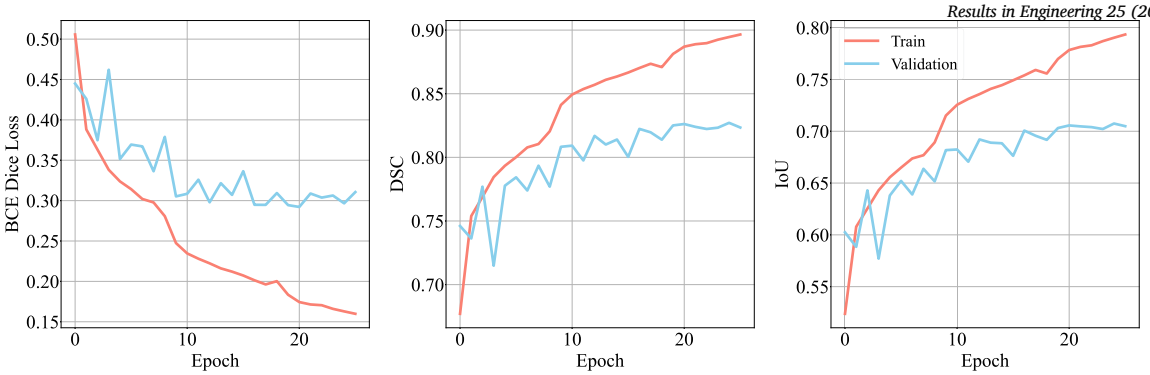

Fig. 6. Training Metrics of Two-Stage Detection and Segmentation Model. From left to right, the BCE Dice loss, DSC, and IoU change curves during the training process of the two-stage detection and segmentation model.

Table 1
The result of segmentation models on SEVERSTAL dataset.

Model	Train Loss	Val Loss	Train DSC	Val DSC	Train IoU	Val IoU
U-Net with EfficientNet-B0	**0.26**	**0.32**	**0.78**	**0.72**	**0.56**	**0.52**
U-Net with Resnet34	0.30	0.35	0.77	0.68	0.50	0.48
FCN-8	0.40	0.41	0.64	0.63	0.40	0.38
FCN-8 with VGG16	0.37	0.39	0.69	0.66	0.48	0.45
FPN with EfficientNet-B0	0.27	0.33	0.76	0.71	0.56	0.51
FPN with Resnet34	0.35	0.36	0.72	0.68	0.51	0.48

are resized to a uniform size of 256×256 pixels to ensure consistency and facilitate efficient processing.

The dataset for the second stage consists of 22,159 cropped images of defected areas. The dataset is divided into 80% for training and 20% for validation. The batch size is set to 32, and the Adam optimizer is used with an initial learning rate of 10^{-3}, which is decayed exponentially during training until it reaches 10^{-11}.

To prevent overfitting, early stopping and model checkpoint callbacks were implemented. The model's validation loss was monitored, and the model with the minimum validation loss was saved. Although the model was set to train for up to 30 epochs, convergence was observed around epoch 20. Due to early stopping, the model effectively trained for 26 epochs, but the best models saved were at epochs 20. This approach ensures that the final model used for evaluation has the best generalization performance, effectively mitigating overfitting.

Fig. 6 illustrates the training and validation loss curves, along with the DSC and IoU metrics over the epochs. Including this figure in the training configuration helps justify the selection of training parameters and demonstrates how overfitting was avoided.

The training parameters and settings for the segmentation model in the second stage are similar to those used in Subsection 4.1.4. Specifically, the number of classes, batch size, optimization algorithm, loss function, and evaluation metrics are the same. By training the model on these cropped images, it can accurately identify and segment the defected areas in the original images.

5. Results and discussion

A comprehensive overview of the results obtained on the desired dataset has been presented. The focus of the results is on the performance evaluation of segmentation models, object detection models, and two-stage detection models. The training of these models is executed using TensorFlow on an NVIDIA TESLA P100 GPU.

5.1. Semantic segmentation

In this part, the results from three different semantic segmentation models are discussed. The U-Net and FPN models are trained using both EfficientNet-B0 and Resnet34 as pre-trained backbones, respectively, while the FCN-8 model uses both the original model and a version with a pre-trained VGG16 backbone.

As detailed in Subsection 4.1.5, the accuracy of each segmentation model is evaluated by DSC and IoU metrics, and the BCE DL function is used as the loss function. Table 1 shows the evaluation results for each model. The results for the U-Net and FPN models with pre-trained backbones are close to each other; however, the U-Net model with an EfficientNet-B0 backbone has the highest scores, with a DSC of 72%, and IoU of 52%.

Comparisons with the results reported in [19] indicate that the U-Net segmentation model with an EfficientNet-B0 backbone achieved a validation DSC of 72% and IoU of 52%, slightly outperforming the U-Net with EfficientNet reported in [19], which achieved a DSC of 70.9% and an IoU of 55.4%. While the DSC is slightly higher, the decrease in IoU can be attributed to a trade-off between the metrics, where a higher DSC may result from the model capturing more predicted regions, but with a few more false positives, leading to a slight reduction in IoU. For the FPN models, the FPN with EfficientNet-B0 backbone achieved a validation DSC of 71% and IoU of 51%, while in [19], the FPN with EfficientNet achieved a DSC of 69.8% and IoU of 53.5%. Again, the higher DSC achieved in this study suggests better overlap between predicted and ground truth masks, but the slightly lower IoU indicates a similar trade-off effect. These comparisons highlight the impact of backbone selection, model architecture, and data distribution on segmentation performance, underlining the subtle balance between DSC and IoU in defect detection tasks.

As discussed in Section 3, the SEVERSTAL dataset comprises four distinct categories of defects: pitted surface, crazing, scratch, and patch defects. Fig. 7 illustrates the prediction results of the U-Net model with an EfficientNet-B0 backbone, which is the optimal segmentation model, on the validation set images. Each image includes the ground truth with a border, along with the predicted image that uses a unique color assigned to its corresponding defect.

The results shown in Fig. 7 demonstrate that the U-Net model with an EfficientNet-B0 backbone provides more accurate predictions for defects in the third and fourth classes (i.e., scratch and patch defects) compared to the first and second classes (i.e., pitted surface and crazing defects). For the first class, this disparity may be attributed to the small size of the defects, making them more challenging for the model to detect and

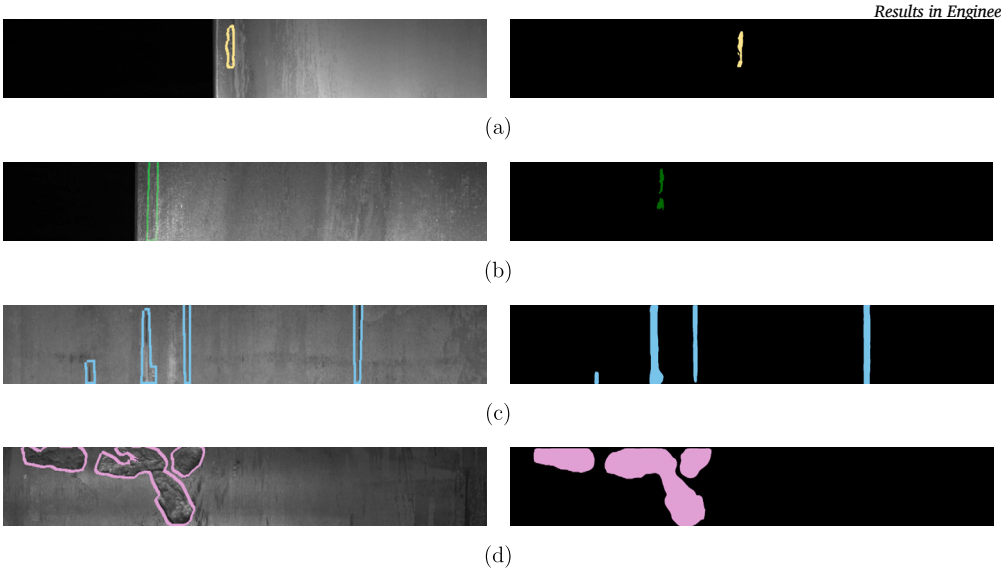

Fig. 7. Prediction results of the U-Net with EfficientNet-B0 backbone model. (a) Pitted surface defect marked by yellow, shows acceptable performance of the model, (b) Crazing defect marked by green, shows poor performance of the model, (c) Scratch defect marked by blue, shows accurate performance of the model, (d) Patch defect marked by purple, shows accurate performance of the model.

Table 2
The results of object detection on SEVERSTAL dataset.

Model	Resulotion	Train Total Loss	Val Total Loss	Train mAP %	Val mAP %
Binary Class	416	1.75	2.21	58.93	44.09
	544	1.68	2.18	60.03	45.1
	608	1.56	2.11	61.22	47.81
	640	**1.44**	**1.99**	**62.9**	**49.32**
Multi-Class	416	3.54	4.09	39.16	31.85
	544	3.4	3.97	40.96	32.98
	608	3.31	3.92	41.48	34.37
	640	**3.19**	**3.86**	**43.86**	**35.06**

predict accurately. In the case of the second class, the imbalanced distribution of the dataset and the limited number of samples likely contribute to the reduced prediction accuracy. As mentioned in Section 3, scratch defects constitute 41% of the entire dataset, and their significant contribution leads to more precise predictions. On the other hand, patch defects have a larger size than other defect types, enabling the model to detect and predict them more accurately. Additionally, the resulting model can accurately predict all defected areas of scratch and patch defects, while pitted surface defects and crazing defects have a smaller contribution to the dataset and are smaller in size compared to scratch and patch defects.

The U-Net architecture is specifically designed to effectively segment small objects and recover spatial information that may be lost during the down-sampling process. Therefore, it can be considered a robust model for accurately identifying and segmenting small defects present on the steel surface. In addition, U-Net has skip connections between the encoder and decoder layers, which helps to recover fine details of the defects. Moreover, the U-Net and FCN-8 architectures are specifically designed for segmentation tasks, which makes them more suitable for this type of task compared to the FPN models which are primarily designed for object detection tasks. So, U-Net's architecture and design could be the reason why it outperforms FCN-8 and FPN on the SEVERSTAL dataset.

5.2. Object detection

Two different approaches are employed for object detection YOLOv4 model: binary class and multi-class object detection. In the binary class approach, the model was designed to detect any type of defect present in the image, regardless of the specific defect type. On the other hand, in the multi-class approach, the model was trained to detect four different types of defects.

Table 3
Object Detection results for each class in model with input resolution of 640.

Class	Train mAP%	Val mAP%
Pitted	42.2	35.87
Crazing	18.53	11.45
Scratch	**59.16**	**50.16**
Patch	50.54	42.74
Total	43.86	35.05

The accuracy of the object detection model is evaluated using the mAP metric, as described in Subsection 4.1.5. Generally, models with higher input resolutions tend to have higher mAP scores. As Table 2 shows, in the binary class approach, the model with resolutions of 608 and 640 achieved similar results, but the model with a resolution of 640 had the highest mAP of 49.32%. Similarly, in the multi-class approach, the models with resolutions of 608 and 640 had comparable results, but the model with a resolution of 640 had a higher mAP of 35.06%.

Table 3 presents the AP values for each type of defect in the multi-class model with a resolution of 640. Scratch and patch defects have been detected with higher accuracy than the other defect types, as the AP values for these classes are 50.16% and 42.74%, respectively. These results are attributed to the larger contribution of scratch defects in the dataset and the larger size of patch defects compared to other defects, allowing the model to detect them more accurately. The class of crazing

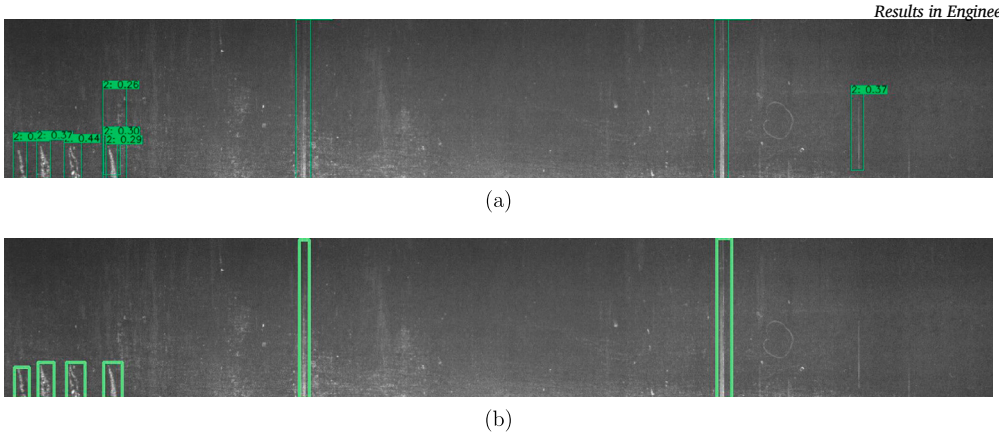

Fig. 8. Predicted Results of YOLOv4 on SEVERSTAL Dataset. Predicted results after training the SEVERSTAL dataset with YOLOv4: (a) the predicted image and (b) the ground truth.

Table 4
The result of two-stage detection and segmentation on SEVERSTAL dataset.

Segmentation Backbone	Train loss	Val Loss	Train Dice	Val Dice	Train IOU	Val IOU
EfficientNet-B0	0.17	0.29	0.89	0.84	0.78	0.71

defects has the lowest AP value of 11.45%, likely due to the smaller number of images in this class compared to other defect categories.

Furthermore, Fig. 8 illustrates the predicted bounding boxes of the corresponding multi-class model on a selected image from the validation set, where the image consists solely of defects belonging to the scratch defects class (denoted as class two in Fig. 8, as classes are indexed from zero to three). As shown in Fig. 8, the model detected most of the defects. However, some false positive regions were identified, which are not marked in the ground truth but exhibit features similar to actual defects. These may be due to the surface texture, not represented by the annotations in the ground truth. Although they bear visual resemblance to actual defects, their inclusion as false positives lowers the mAP score. These few more detections, though correctly predicting the true defects, actually lower the value of general performance metrics.

5.3. Two-stage defect detection

This section presents the outcomes of the second stage of the proposed method in Subsection 4.3, which involves using a segmentation model to identify defected areas within the cropped images. The U-Net model with an EfficientNet-B0 backbone is employed, which is determined to be the best segmentation model in Subsection 5.1.

Table 4 presents the training outcomes of the proposed segmentation model using U-Net with EfficientNet-B0 backbone. A comparison of the results in this table with the results of U-Net with the EfficientNet-B0 backbone in Table 1 and the experiments in [19] indicates an improvement in segmentation accuracy achieved by the proposed method. The proposed method yields a DSC and IoU of 84% and 71%, respectively, which demonstrate a 12% increase in DSC and a 20% increase in IoU compared to the previous segmentation model.

In the proposed method, the increase in performance of the segmentation model can be attributed to several factors. Firstly, the defected area is obtained at a higher resolution by cropping it from the original image, which provides more detailed information for the segmentation model to work with. Secondly, by specifying the defected area, the segmentation model can focus on the specific region of interest, which leads to more precise segmentation. Moreover, the proposed method reduces the impact of defect size on the segmentation accuracy, while maintaining the significant influence of the number of images for crazing defects in the second class. This results in an overall improvement in the segmentation performance, as demonstrated by the higher DSC and IoU scores in Table 4 compared to Table 1.

As previously described in Subsection 5.1, the dataset used in this study includes different defect categories such as pitted surface, crazing, scratch, and patch defects. Following the cropping of the defected areas, both pitted surface and crazing defects were found to have similar size scales and were provided in the same resolution. Therefore, the corresponding segmentation model for both of these defects can be classified into the same category. Fig. 9 presents the prediction results of the segmentation model in the two-stage detection approach, obtained using images from the validation subset of the dataset. The figure illustrates three distinct categories, where pitted surface and crazing defects are marked by yellow and green, respectively, and are classified into the first category, while the scratch defect is marked with blue and the patch defect is marked with purple and are classified into the second and third categories, respectively.

Fig. 9 displays the prediction results of the segmentation model of the two-stage detection model, which were obtained using images from the validation subset of the dataset. As depicted in Fig. 9, there is a noteworthy enhancement in the prediction of segmented areas for pitted surface and crazing defects compared to the results shown in Fig. 7. While there are still some instances of misprediction in this category, overall, the performance has significantly improved. Conversely, for both scratch and patch defects, the corresponding model accurately predicts the segmented areas, even more accurately than the segmentation model in Subsection 5.1. It is worth noting that the large proportion of scratch defects in the dataset and the comparatively larger size of patch defects contribute to the accurate prediction of these types of defects by the model. Thus, the proposed method mainly addresses the issue of small contribution and size of the pitted surface and crazing defects, with less impact on the prediction of scratch and patch defects.

6. Conclusion

This study presented a comprehensive approach for detecting and segmenting defects on steel surfaces using deep-learning techniques, specifically addressing challenges associated with imbalanced datasets and small defect sizes. By evaluating three different segmentation models—U-Net, FCN-8, and FPN—with pre-trained backbones on the SEVERSTAL dataset, it was found that the U-Net model with an EfficientNet-B0 backbone achieved the highest accuracy.

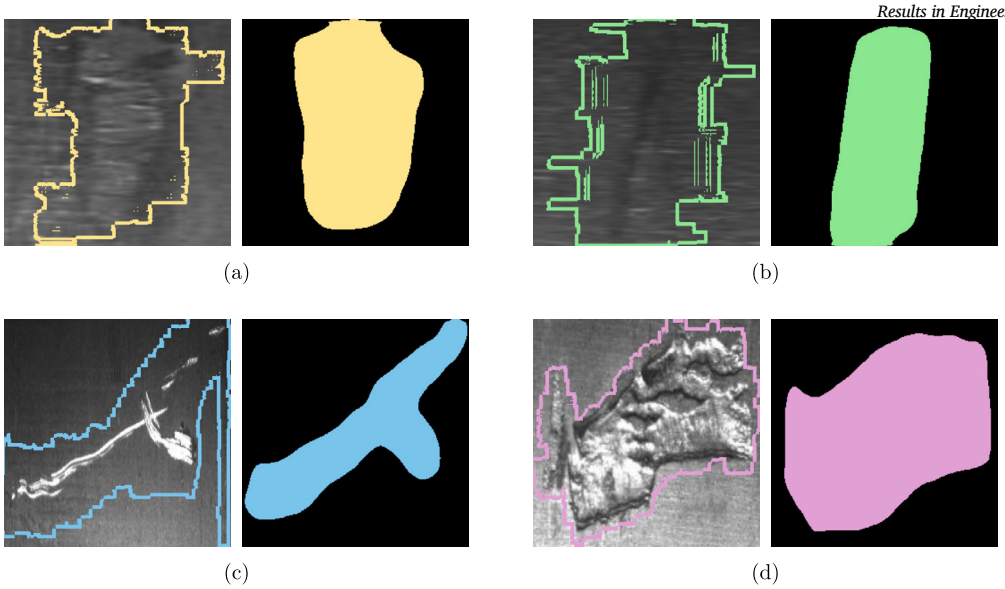

Fig. 9. Prediction results of the second stage of the proposed method using the U-Net with EfficientNet-B0 backbone model. (a) Pitted surface and (b) crazing defects marked by yellow and green, respectively, show a significant improvement in the model's performance. (c) Scratch defect marked by blue shows more accurate performance. (d) Patch defect marked by purple shows better performance.

Comparing the results with recent studies, the U-Net segmentation model with an EfficientNet-B0 backbone achieves a higher DSC than the U-Net with EfficientNet in [19], demonstrating the effectiveness of this approach. While the FPN model achieved slightly lower performance than that reported in [19], the overall results validate the efficient performance of the studied techniques for accurate defect detection on steel surfaces.

To enhance the detection and segmentation of small and underrepresented defect classes, a novel two-stage approach combining object detection and segmentation was proposed. This method involved using YOLOv4 for initial defect detection, followed by U-Net for precise segmentation of the cropped defect areas. The two-stage approach significantly improved the DSC to 84% and the IoU to 71%, outperforming the single-stage models and effectively addressing the issues caused by imbalanced data and varying defect sizes.

The results demonstrate that the proposed method effectively improves the detection accuracy of small and less frequent defects, which are often challenging due to their limited representation in the dataset. By focusing on these challenges, the approach contributes to more reliable and comprehensive defect detection systems for steel surfaces. Future work may involve further balancing the dataset through data augmentation or exploring additional techniques to enhance the detection of underrepresented defect classes.

Overall, the study highlights the potential of combining object detection and segmentation models to address specific challenges in defect detection tasks, providing valuable insights for developing more effective inspection systems in the manufacturing industry.

CRediT authorship contribution statement

Sara Ashrafi: Writing – original draft, Writing – review & editing. **Sobhan Teymouri:** Writing – original draft, Writing – review & editing. **Sepideh Etaati:** Writing – original draft, Writing – review & editing. **Javad Khoramdel:** Supervision. **Yasamin Borhani:** Supervision. **Esmaeil Najafi:** Writing – review & editing, Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The data used are publicly available on the Kaggle website and are referenced in the reference section.

References

[1] I. Baillie, P. Griffith, X. Jian, S. Dixon, Implementing an ultrasonic inspection system to find surface and internal defects in hot, moving steel using emats, Insight 49 (2007) 87–92, https://doi.org/10.1784/insi.2007.49.2.87.

[2] J.-l. Lu, G.-g. Cheng, M. Wu, et al., Detection and analysis of magnetic particle testing defects on heavy truck crankshaft manufactured by microalloyed medium-carbon forging steel, J. Iron Steel Res. Int. 27 (2020) 608–616, https://doi.org/10.1007/s42243-019-00334-7.

[3] K. Manikandan, P. Ashwin Sivagurunathan, S. Ananthan, A. Arul Marcel Moshi, S. Sundara Bharathi, Study on the influence of temperature and vibration on indications of liquid penetrant testing of a516 low carbon steel, in: 2nd International Conference on Recent Trends in Metallurgy, Materials Science and Manufacturing, Mater. Today Proc. 39 (2021) 1559–1564, https://doi.org/10.1016/j.matpr.2020.05.572, https://www.sciencedirect.com/science/article/pii/S2214785320341754.

[4] F. Akhyar, Y. Liu, C.-Y. Hsu, T.-K. Shih, C.-Y. Lin, Fdd: a deep learning-based steel defect detectors, Int. J. Adv. Manuf. Technol. 126 (2023) 1093–1107, https://doi.org/10.1007/s00170-023-11087-9.

[5] O. Ronneberger, P. Fischer, T. Brox, U-net: convolutional networks for biomedical image segmentation, https://arxiv.org/abs/1505.04597, arXiv:1505.04597, 2015.

[6] J. Long, E. Shelhamer, T. Darrell, Fully convolutional networks for semantic segmentation, https://arxiv.org/abs/1411.4038, arXiv:1411.4038, 2015.

[7] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, S. Belongie, Feature pyramid networks for object detection, https://arxiv.org/abs/1612.03144, arXiv:1612.03144, 2017.

[8] Kaggle, Severstal: steel defect detection, https://kaggle.com/competitions/severstal-steel-defect-detection, 2019.

[9] A. Bochkovskiy, C.-Y. Wang, H.-Y.M. Liao, Yolov4: optimal speed and accuracy of object detection, https://arxiv.org/abs/2004.10934, arXiv:2004.10934, 2020.

[10] Y. Chung, R. Shrestha, S. Lee, W. Kim, Thermographic inspection of internal defects in steel structures: analysis of signal processing techniques in pulsed thermography, Sensors 20 (2020), https://doi.org/10.3390/s20216015, https://www.mdpi.com/1424-8220/20/21/6015.

[11] W. Chen, Y. Gao, L. Gao, X. Li, A new ensemble approach based on deep convolutional neural networks for steel surface defect classification, in: 51st CIRP Conference on Manufacturing Systems, Proc. CIRP 72 (2018) 1069–1072, https://doi.org/10.1016/j.procir.2018.03.264, https://www.sciencedirect.com/science/article/pii/S2212827118304359.

[12] A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, in: F. Pereira, C. Burges, L. Bottou, K. Weinberger (Eds.), Advances in Neural Information Processing Systems, vol. 25, Curran Associates, Inc., 2012, pp. 1097–1105, https://proceedings.neurips.cc/paper_files/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf.

[13] A. Komijani, F. Vafaeinezhad, J. Khoramdel, Y. Borhani, E. Najafi, Multi-label classification of steel surface defects using transfer learning and vision transformer, in: 2022 13th International Conference on Information and Knowledge Technology (IKT), 2022, pp. 1–5.

[14] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C.L. Zitnick, P. Dollár, Microsoft coco: common objects in context, https://arxiv.org/abs/1405.0312, 2015, arXiv:1405.0312.

[15] I. Konovalenko, P. Maruschak, J. Brezinová, O. Prentkovskis, J. Brezina, Research of u-net-based cnn architectures for metal surface defect detection, Machines 10 (2022), https://doi.org/10.3390/machines10050327, https://www.mdpi.com/2075-1702/10/5/327.

[16] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, https://arxiv.org/abs/1512.03385, arXiv:1512.03385, 2015.

[17] I. Konovalenko, P. Maruschak, H. Kozbur, J. Brezinová, J. Brezina, B. Nazarevich, Y. Shkira, Influence of uneven lighting on quantitative indicators of surface defects, Machines 10 (2022), https://doi.org/10.3390/machines10030194, https://www.mdpi.com/2075-1702/10/3/194.

[18] P. Damacharla, A.R.M. V., J. Ringenberg, A.Y. Javaid, Tlu-net: a deep learning approach for automatic steel surface defect detection, https://arxiv.org/abs/2101.06915, arXiv:2101.06915, 2021.

[19] Z. Aboulhosn, A. Musamih, K. Salah, R. Jayaraman, M. Omar, Z. Aung, Detection of manufacturing defects in steel using deep learning with explainable artificial intelligence, IEEE Access 12 (2024) 99240–99257, https://doi.org/10.1109/ACCESS.2024.3430113.

[20] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, A.L. Yuille, Deeplab: semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs, IEEE Trans. Pattern Anal. Mach. Intell. 40 (2018) 834–848, https://doi.org/10.1109/TPAMI.2017.2699184.

[21] M. Mirza, S. Osindero, Conditional generative adversarial nets, arXiv preprint, arXiv:1411.1784, 2014.

[22] Z. Nie, J. Xu, S. Zhang, Analysis on deeplabv3+ performance for automatic steel defects detection, https://arxiv.org/abs/2004.04822, arXiv:2004.04822, 2020.

[23] G. Zhang, Y. Pan, L. Zhang, Semi-supervised learning with gan for automatic defect detection from images, Autom. Constr. 128 (2021) 103764, https://doi.org/10.1016/j.autcon.2021.103764, https://www.sciencedirect.com/science/article/pii/S0926580521002156.

[24] Z. Mi, R. Chen, S. Zhao, Research on steel rail surface defects detection based on improved yolov4 network, Front. Neurorobot. 17 (2023) 1119896, https://doi.org/10.3389/fnbot.2023.1119896.

[25] Y. He, K. Song, Q. Meng, Y. Yan, An end-to-end steel surface defect detection approach via fusing multiple hierarchical features, IEEE Trans. Instrum. Meas. 69 (2020) 1493–1504, https://doi.org/10.1109/TIM.2019.2915404.

[26] D. Weimer, B. Scholz-Reiter, M. Shpitalni, Design of deep convolutional neural network architectures for automated feature extraction in industrial inspection, CIRP Ann. 65 (2016) 417–420, https://doi.org/10.1016/j.cirp.2016.04.072, https://www.sciencedirect.com/science/article/pii/S0007850616300725.

[27] J. Božič, D. Tabernik, D. Skočaj, End-to-end training of a two-stage neural network for defect detection, https://arxiv.org/abs/2007.07676, arXiv:2007.07676, 2020.

[28] S. Etaati, J. Khoramdel, E. Najafi, Automated wheat disease detection using deep learning: an object detection and classification approach, in: 2023 11th RSI International Conference on Robotics and Mechatronics (ICRoM), 2023, pp. 116–121.

[29] D. Tabernik, S. Šela, J. Skvarč, D. Skočaj, Segmentation-based deep-learning approach for surface-defect detection, J. Intell. Manuf. 31 (2019) 759–776, https://doi.org/10.1007/s10845-019-01476-x.

[30] Q. Zhou, R. Chen, B. Huang, C. Liu, J. Yu, X. Yu, An automatic surface defect inspection system for automobiles using machine vision methods, Sensors 19 (2019), https://doi.org/10.3390/s19030644, https://www.mdpi.com/1424-8220/19/3/644.

[31] M.A. Rahman, Y. Wang, Optimizing intersection-over-union in deep neural networks for image segmentation, in: G. Bebis, R. Boyle, B. Parvin, D. Koracin, F. Porikli, S. Skaff, A. Entezari, J. Min, D. Iwai, A. Sadagic, C. Scheidegger, T. Isenberg (Eds.), Advances in Visual Computing, Springer International Publishing, Cham, 2016, pp. 234–244.

[32] K.H. Zou, S.K. Warfield, A. Bharatha, C.M. Tempany, M.R. Kaus, S.J. Haker, W.M. Wells, F.A. Jolesz, R. Kikinis, Statistical validation of image segmentation quality based on a spatial overlap index, Acad. Radiol. 11 (2004) 178–189, https://doi.org/10.1016/s1076-6332(03)00671-8.

[33] C.H. Sudre, W. Li, T. Vercauteren, S. Ourselin, M. Jorge Cardoso, Generalised Dice Overlap as a Deep Learning Loss Function for Highly Unbalanced Segmentations, Springer International Publishing, 2017, pp. 240–248.

[34] I. Goodfellow, Y. Bengio, A. Courville, Deep Learning, MIT Press, 2016, http://www.deeplearningbook.org.

[35] S. Jadon, A survey of loss functions for semantic segmentation, in: 2020 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB), 2020, pp. 1–7.

[36] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, C.L. Zitnick, Microsoft coco: common objects in context, in: D. Fleet, T. Pajdla, B. Schiele, T. Tuytelaars (Eds.), Computer Vision – ECCV 2014, Springer International Publishing, Cham, 2014, pp. 740–755.