Hindawi
Complexity
Volume 2022, Article ID 4637939, 17 pages
https://doi.org/10.1155/2022/4637939

Research Article

Real-Time Explainable Multiclass Object Detection for Quality Assessment in 2-Dimensional Radiography Images

Sadra Naddaf-Sh,[1] M-Mahdi Naddaf-Sh,[1] Hassan Zargarzadeh,[1] Maxim Dalton,[2] Soodabeh Ramezani,[2] Gabriel Elpers,[2] Vinay S. Baburao,[2] and Amir R. Kashani[2]

[1]*Phillip M. Drayer Electrical Engineering Department, Lamar University, Beaumont, TX, USA*
[2]*Artificial Intelligence Lab, Stanley Oil & Gas, Stanley Black & Decker, New Britain, CT, USA*

Correspondence should be addressed to Hassan Zargarzadeh; hzargarzadeh@lamar.edu

Received 29 May 2021; Revised 22 August 2021; Accepted 30 August 2021; Published 8 August 2022

Academic Editor: Long Wang

Quality inspection and defect detection play a critical role in infrastructure safety and integrity specially when it comes to aging infrastructure mostly owned by governments around the world. One of the prevalent inspections performed in the industry is nondestructive testing (NDT) using radiography imaging. Growing demand, shortage of experts, diversity of required skills, and specific regional standards with a time-limited requirement of inspection results make automated inspection an urgent need. Therefore, utilizing artificial intelligence- (AI-) based tools as an assistive technology has become a trend for industrial applications, which automates repeated tasks and provides increased confidence before and during the inspection operation. Most of the works in quality assessment are focused on the classification of few categories of defects and mostly performed on public or noncomprehensive research datasets. In this work, a scalable, efficient, and real-time deep learning family of models for detection and classification of 10 various categories of weld characteristics on a real-world industrial dataset is presented. The models are evaluated and compared against each other, various critical hyperparameters and components are optimized, and local explainability of models is discussed. Additionally, AutoAugment for object detection and various techniques are utilized and investigated. The best performance for object detection and classification for 10 class models is reached by mean average precision of 72.4% and top-1 accuracy of 90.2%, respectively. Also, the fastest object detection model is able to evaluate a full 15360×1024 pixels weld image in 0.39 seconds. Finally, the proposed models are deployable on edge-devices to perform as assistant to NDT experts or auditing professionals.

1. Introduction

Inspection and assessment of welded joints are critical in many industries such as marine, aerospace, and chemical, and specifically in oil and gas industries [1]. Welded joints are among vulnerable parts of any industrial infrastructure including pipelines. Hence, preliminary weld inspection during the construction has a crucial role in its longevity as a small discontinuity can grow into an utter failure over time [2, 3]. Moreover, pipeline failures can damage life in large-scale and is a threat to the environment [4]. Furthermore, it is very costly to maintain continuous inspection to track the growth of initial imperfections over time or efforts to restore the surrounding environment or the pipeline when defects are larger than a certain threshold [5, 6]. Thus, weld inspection is the most economical preventive approach specifically at early stages of its construction. Among different nondestructive testing (NDT) technologies at the point of constructions, radiographic testing (RT), ultrasonic testing (UT), and magnetic testing (MT) are of great importance. Currently RT, in which X-ray imaging of the welded part is done, is preferred due to the universal training and accuracy of its technology [7]. Nonetheless, analysis of X-ray images is time-consuming and tedious, and at the end different experts might have different opinions and hence auditing is essential [8]. Thus, automation of these systems is of interest in the industry to certify reliability and safety of the product in various stages of construction, approval, audit, and risk assessment.

In recent decades, many research have been conducted on automation of tasks employing robots [9–12], including robotic platforms automation of welding operation to accelerate the process and reduce human error. As an instance, Figure 1 shows a robotic digital X-ray photographer by Stanley Oil and Gas. The robot autonomously conducts the X-ray imaging that significantly minimizes the human intervention to prevent the operators from exposure [13]. After a robotic imaging process is done, human experts use images generated to inspect the welds. However, recent rapid improvement in machine learning, computer vision, and pattern recognition has opened new roads to provide novel solutions in order to address the challenges regarding ultimate defect diagnosis and complete tractability of discontinuities over the pipeline's life cycle [2, 3, 8, 14, 15]. In the following, a review on related research performed in weld and defect diagnosis is provided.

Previous research works with focus on defect analysis are mainly divided into two smaller subgroups. Before the prevalence of deep learning and convolutional neural networks (CNNs) approaches in the early 2010s, procedures focused on traditional image processing methods for image preprocessing and classification utilizing classical machine learning methods (e.g., support vector machine (SVM)) and training artificial neural networks (ANNs) based on hand-crafted features extracted from image patches (cropped rectangular pieces of a larger image). Among these works, they mainly focused on classifying defect and nondefect images and assigning a single label to an image patch with or without segmentation of defect area. Mery and Berti [16] used texture features to train ANNs and the best result reached 8% false alarm. In [17], gray level co-occurrence matrix (GLCM) texture features were used for multiclass ANNs with 86.1% accuracy and optimized to reach 87.3% by applying Levenberg–Marquardt optimizing function in [18]. A similar approach to classify defects with a combination of statistical and geometric features and utilizing top-hat filtering, thresholding, and morphological smoothing as preprocessing presented in [19] resulted in 91% accuracy in detecting defects and nondefects and 96% in classifying of a hundred of test images containing low contrast images. In [20], Wiener filter is considered the best enhancement as it leads to lower rooted mean square error (RMSE) in comparison with median filtering and contrast enhancement, and also defective segments are obtained from the segmented image using an automatic threshold. Finally, for feature extraction, the lexicographically-ordered one-dimensional signal of the image is generated, and mel-frequency cepstral coefficients (MFCCs) and polynomial coefficients are extracted from the power density spectra (PDSs) of the image and passed into ANN, which reduced false positive rate to 7%. Lim al. [21] employed a multilayer perceptron (MLP) network trained on a simulated dataset of weld radiographic images for classification of the patches.

Zapata et al. in [22] used an adaptive network-based fuzzy inference system (ANFIS) and ANN, in which geometrical and texture features were selected with respect to minimizing computational complexity and reached 82.6% accuracy. Valavanis and Kosmopoulos [23] applied certain classifiers for distinguishing between six types of defects annotated based on British Standards or labeling as nondefect. Preprocessing steps of their research include utilizing local threshold, graph-based segmentation, and then geometric and texture features are used as input for classifiers like ANN, K-nearest neighbor (KNN), and SVM. In [24], a comprehensive review of similar methods is provided. It can be concluded that classical approaches require major preprocessing steps before feature extraction and preprocessing enhancements have direct impact on final accuracy.

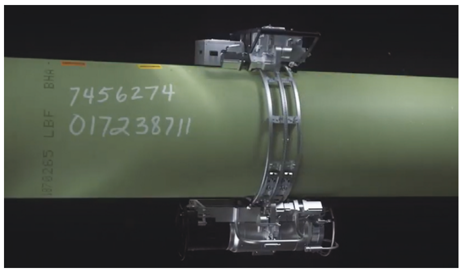


Figure 1: Digital X-ray detector and source on a robotic platform, Stanley Oil and Gas.

On the other hand, a few researches focused on image segmentation to provide a general understanding of defect localization. Carrasco and Mery [25] presented a method for segmenting defects. The method consists of a few steps: median filtering, bottom-hat filter, binary thresholding, and watershed transform. The results suggested an area under curve (AUC) of 93.58% for ten images. In [26], sliding window approach is used for weld object detection based on a large set of features. In [27], Ben Gharsallah and Ben Braiek proposed a method to address nonrobustness of defect segmentation caused by uneven illumination, based on level set active contour guided with an off-center saliency map, in which an energy function gets minimized to achieve segmentation. Despite faster convergence and higher accuracy than local image filtering and contrast enhancement, the method requires further investigation to minimize human intervention in finding region of interest (ROI). In [28], defect segmentation problem is addressed using Gabor filtering and canny edge detector. As more recent research, which is also evaluated on aerospace weld dataset, a novel pixelwise segmentation defect detection system is presented in [8]. Dong et al. [8] described a system to detect weld defects by using random forest instead of Softmax as the classifier of a U-net [29]. The approach is pixelwise labeling of highly similar circular defects, which are prevalent in aerospace industries.

Since the prevalence of deep convolutional neural networks (DCNNs), many works have focused on using these models for feature extraction/selection instead of traditional hand-crafted feature extraction and nonrobust methods. Primarily two general tasks are performed using DCNNs (i.e., classification and object detection task). Furthermore, weld defect dataset has class-imbalance issue, since the number of weld defects might not distribute equally among different classes. Hoe et al. [30] focused on extending three types of datasets using auto encoders to address the imbalance problem. Next, a few models, including DCNNs and other models based on extracted features are trained to classify four different types of defects and reached accuracy of 97.2%. Ajmi et al. [31] explored two-class (porosity and lack of penetration) classification of weld defects. Data augmentation through horizontal mirroring, translations, and RGB channels modification are applied to boost model performance, and 85.2% accuracy is reported with transfer learning utilizing AlexNet [32] and addition of a few dropout layers as well as modified final layer on GDXray [33]. In [34], a real-time and two-stage method based on images from a 3D laser scanner is proposed. The method performs four-class classification of narrow lap welds. Also, a comparison on classical and deep classification methods is performed with average accuracy of 80% for classical approaches while for deep methods of VGG-16 [35] and ResNet50 [36], 97.1% and 97.8% accuracy are reported, respectively. Wang et al. [37] presented a tutorial for weld defect detection based on DCNNs with implementation provided in PyTorch [38]. The paper provides a step-by-step approach for the data collection, preprocessing, and model designing, training, and testing.

Further investigation is performed for accurate localization of weld characteristics using deep methods. Hou et al. [14] designed a deep learning-based system for weld quality assessment. They used sparse autoencoder (SAE) to extract and use intrinsic features for classifying 32×32 pixels weld patches and finally using a sliding window to classify image pixels as defect or nondefect. The process reaches an accuracy of 91% on GDXray [33], even though the work is a binary class defect classification and the process is time-consuming because of the nature of the sliding window approach and size of full weld images. In [39], extensive experiments with 24 various computer vision-based weld object detection methods (including deep learning methods based on sliding window) are performed and reported. In [40], two-stage detectors (i.e., Faster RCNN [41]) are used whose task is object detection of weld defects in shipbuilding which accounts for 60% of the building process, where radiography testing is used to inspect welded joints. The proposed object detector is trained to detect two general types of porosity and lack of fusion/slag defects. Moreover, the best result is acquired by data augmentation, which reached 53.2 mean average precision (mAP) on Faster RCNN [41] with ResNet50 [36] backbone.

Gau et al. [42] developed a contrast enhancement conditional generative adversarial network (GAN) to address the contrast and class-imbalance issue. There are two separate target networks in their work. The first network accepts a 71×71 pixels patch from weld seam to classify the patch as defect/nondefect. For determining defect type, defective patches are passed into a second classification network. At the end, the sliding window approach is used for localizing defects. Thus, with respect to the two-stage design of the system and the sliding window, the entire system will not perform in real-time for high-resolution images. In [43], a defect localization method based on U-net and augmentation using conditional GAN (cGAN) [44] is presented, and the method is evaluated on GDXray dataset [33]. Although the method shows AUC of 88.4% for defect segmentation, lack of defect classification is discernible. Gantala and Balasubramaniam [45] presented an automatic defect recognition model trained on total focusing method (TFM) imaging dataset and finite element simulated dataset with addition of noise and further expansion of dataset utilizing deep convolutional gaN (DCGAN). Their two-class defect detection model was evaluated with yolov4 [46] and reached 85 average precision (AP) on the noisy dataset.

Although the above research papers are mostly related to employing deep CNN methods to automate the preliminary inspection in construction and welding, studies using deep CNN methods for NDT and defect diagnosis are not limited to radiography images and weld construction. Yan et al. [47] developed deep models for enhanced feature extraction and ultrasonic pattern recognition for inspection gas pipelines. The method uses contact-less dual-mode bulk wave electromagnetic acoustic transduce (EMAT) and interpretations of A-scan signals to detect defects. It leverages continuous wavelet transform (CWT) to extract frequency-time domain features, then a deep CNN model is applied to perform high-end feature extraction, and finally, a pretrained SVM is used for defect/nondefect classification of signals. The method feature extraction ability is verified by comparing to other methods, including discrete wavelet transform (DWT) and statistical features, all of which are outperformed by the CNN model, which achieves 93.75% accuracy on a dataset of pipe with artificially manufactured defects. The work is performed for defect/nondefect classification, and the possibility of defect type classification is to be investigated.

In addition to ultrasonic pattern recognition, deep CNNs are also utilized for thermography crack detection. In [48], Hu et al. explored supervised thermography video sequence metal crack detection and localization. The work uses eddy current pulsed thermography (ECPT), a multi-physics coupling method, to detect turbulence in conductive materials by analyzing thermal patterns. Initially, principal component analysis (PCA) is used to extract thermal sequence components from original data. Then, Faster RCNN [41] is used to perform object detection on images accurately. Finally, the method is compared to traditional detection methods, and it demonstrates 0.97 probability of detection, which outperforms the accurate prior method by 26%. Proposed methods are validated experimentally and have shown significant improvement in their own type of NDT and data acquisition, demonstrating the advantages of using CNN for feature extraction in NDT. While UT and thermography methods (e.g., ECPT) are commonly used for in-line inspection and maintenance purposes and not for weld construction inspections, these methods have their limitations, such as low sensitivity to small defects or internal crack detection [13].

Studies mentioned above are all experimentally evaluated on either (1) a set of images from a private dataset (i.e., usually created for experimental purposes) or (2) GDXray [33] or similar public and noncomprehensive sets. As shown in Figures 2 and 3, there are noticeable differences in images from welded joints at Stanley, and the GDXray dataset. First, GDXray has a limited number of samples. Second, class

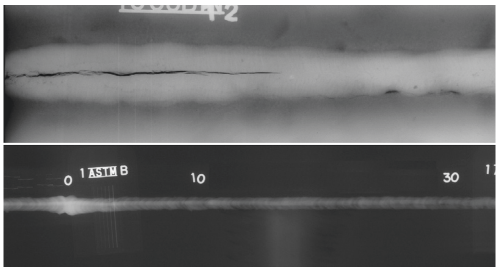


Figure 2: Tow samples of images in the GDXray database (top) and SBD dataset (bottom). The bottom image is cropped to be able to compare with each other. Defects are more visible in the GDXray database than SBD dataset.

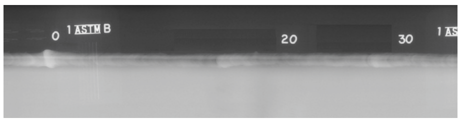


Figure 3: An image sample from SBD database in that two surfaces with different thicknesses are welded.

diversity is limited and also annotations and weld characteristics are based on a different standard [33]. Third, visibility of defects is limited compared to defects at Stanley. Also, in some cases, a single patch contains more than one type of defect, which does not permit the experts to designate a single label for the entire image patch, all of which make classification only or defect/nondefect localization incompatible with real-world industrial requirements and standards. In other words, the detection of non-hand-picked and diverse real-world samples is more of a challenge. On the other hand, since the systems will work as assistant to NDT experts, and there are limitations in hardware for deploying as well as time-constraint processing requirements, scalability is required for efficient and optimized utilization. Considering mentioned reasons, these methods either fail to reach required specifications or do not meet required performance based on industry measures.

This paper aims to address the accuracy and inference time trade-offs by presenting an efficient and scalable set of deep models. Moreover, instead of assigning a single label for each patch, accurate location and label for each discontinuity will be determined. The contributions in this work are as follows: (1) describing an efficient and scalable system for object detection or classification of weld characteristics on long, high-resolution radiography weld images, which is deployable as a real-time assistant for NDT experts, (2) demonstration and analysis of the transferring augmentation strategies during training which can improve the performance of the system on detection of rare small discontinuity which are easier to miss during manual inspection and harder to detect with deep learning methods, (3) analyzing and experimenting with different components of the deep model, such as activation functions, and feature extraction backbones, and (4) comparative analysis on the presented models with base-line models.

The rest of this article is organized as follows. In Sections 2 and 3, an overview of dataset preparation and proposed methods is provided, respectively, as well as description of system architecture. In Section 4, the methods are tested, various models are described, and the augmentation approach and results are evaluated. Finally, conclusions are proposed in Section 5.

2. Dataset

The dataset contains thousands of X-ray images taken with the purpose of NDT of weld construction in preliminary stages. There is little to no material variation in weld construction, which helps developing a model focusing on accuracy and robustness. The majority of the structures are plain carbon steel. The diameter of the pipes ranges from 24 to 56 inches. However, pipes with either 36 inches or 42 inches are mostly common. Moreover, the pipes wall thickness is at least 0.5 inches with the grade of X65 or greater. Finally, all pipes are consistent with API 5L [49] in terms of types, dimensions, material, and grade.

Welded-joint images have various resolutions depending on the exterior diameter of the structure. In this dataset, the resolution of the images is roughly 15360×1024 pixels, with the occurrence of weld discontinuities. As the welded area only covers one-fifth of each weld image's center area, images are cropped into 224×224 patches with 20% overlap. This overlap benefits in two ways. First, it assists in retaining defects lying in between two patches in one patch. Second, as smaller defects shift in two consecutive images, it can be interpreted as data augmentation. Next, experts annotated the images based on API 1104 [50] standards. Most of the defect-free patches are removed from the dataset to prevent overwhelming the network with nondefect images. Finally, Figure 4 shows samples of the dataset, and Table 1 shows the distribution of images for each set. As the dataset reveals, about 75% is used as train set (i.e., 17872 images), and 10% and 15% are used as dev/validation set and test set, respectively. Note that the dataset is collected from welding of various structures and different welding devices. Thus, results obtaining from this dataset can demonstrate the generalizability and robustness of proposed solutions for extensive use as assistant to NDT experts. Figure 5 summarizes preprocessing steps on the dataset. The steps are described in detail in Section 3.1.

3. Method

Addressing robustness, accuracy, and time performance are required for employing a deep convolutional model in production for the task of weld defect object detection. Over recent years, scaling up image resolution, depth and width of the network, and using a larger backbone are widely used to boost the performance of the models [51–54]. However, this costs, in a larger model, higher computation and inference time [51, 52] as well as longer training time. Thus, a robust and efficient design is required to address the accuracy versus time performance. In order to address the trade-off between accuracy and time and achieve efficiency in models, a family of one-stage and scalable models called EfficientDet [52] are exploited. Employing a single compound coefficient,

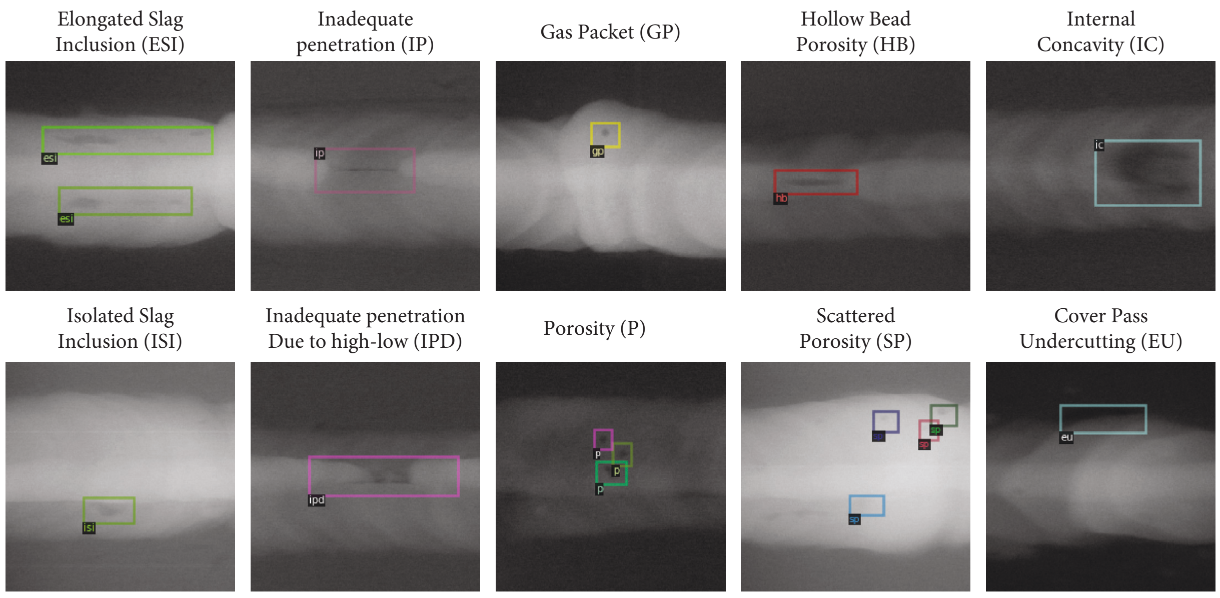

Elongated Slag Inclusion (ESI)
Inadequate penetration (IP)
Gas Packet (GP)
Hollow Bead Porosity (HB)
Internal Concavity (IC)
Isolated Slag Inclusion (ISI)
Inadequate penetration Due to high-low (IPD)
Porosity (P)
Scattered Porosity (SP)
Cover Pass Undercutting (EU)
esi
ip
gp
hb
ic
isi
ipd
p
sp
eu

Figure 4: Samples of classes in dataset.

Table 1: Distribution of images and labels.

	Total	Categories									
		ESI	ISI	IP	IPD	GP	P	HB	SP	IC	EU
Train	17872	4424	2948	908	702	1840	598	4062	498	597	1295
Validation	2203	569	338	118	97	280	71	435	51	87	157
Test	3394	891	490	190	139	408	116	707	83	130	240
Total	23469	5884	3776	1216	938	2528	785	5204	632	814	1692

one can scale the architecture to address the trade-off between model size and accuracy of the model, resulting in a model deployable on various end-devices ranging from mobile devices to high-performance GPU clusters.

A two-stage object detection model generally starts with a search on regions of interest (ROI) using the selective search or, in more recent designs, applying region proposal networks (RPNs), and then by passing image to the second stage for feature extraction, classification of the boxes, and refinement of the bounding box are performed [41, 55]. Although the tow-stage methods might lead to higher accuracy, the inference time because of the first stage burden is significant in sight of the additional step (RPN). In contrast, one-stage detectors apply a feature extractor called backbone and then fuse multilevel extracted features. In the end, class/box networks help to extract class labels and regression of bounding boxes. Since the image passes only once through the network, the one-stage detector performs significantly faster than other methods [54]. By utilizing pretrained backbones, the power from classification tasks transfers to these object detectors as employed in [56]. In this section, preprocessing steps, EfficientDet architecture design, and augmentation strategies for object detection as well as system architecture to achieve an accurate model with low time latency are discussed, respectively.

3.1. System Architecture. Figure 5 depicts the required preprocessing steps to generate the dataset, which start with downloading image patches and quality validation. Although the images on the cloud storage are prevalidated for quality, it can be done through a wire IQI tag, which is discernible on the image in Figure 6. As this step is optional and can be done upon uploading the images to the cloud storage, its time burden is disregarded from total system time performance. As the final two steps, brightness correction and contrast leveling as well as slicing of the original 15360×1024 pixels image with 20% overlap are done.

As Figure 5 training depicts, training starts on a scaled model, which depends on a single coefficient for the determination of depth and width of the network. In addition, AutoAugment during training is performed. Procedures of network design, scaling, and augmentation are elaborated in Sections 3.2–3.4. As the next step, based on the type of the trained network in the model, it predicts either label and accurate location of the defects or assigns a single label for the whole patch with explanations on the decision provided. Finally, Figure 5 visualization indicates stitching as the first step of visualization. Since exact slicing points are saved during slicing, relative predicted defect locations of the whole image are calculable. Finally, the full DICONDE image can be visualized through Stanley web-app or mobile-app or saved as DICONDE metadata.

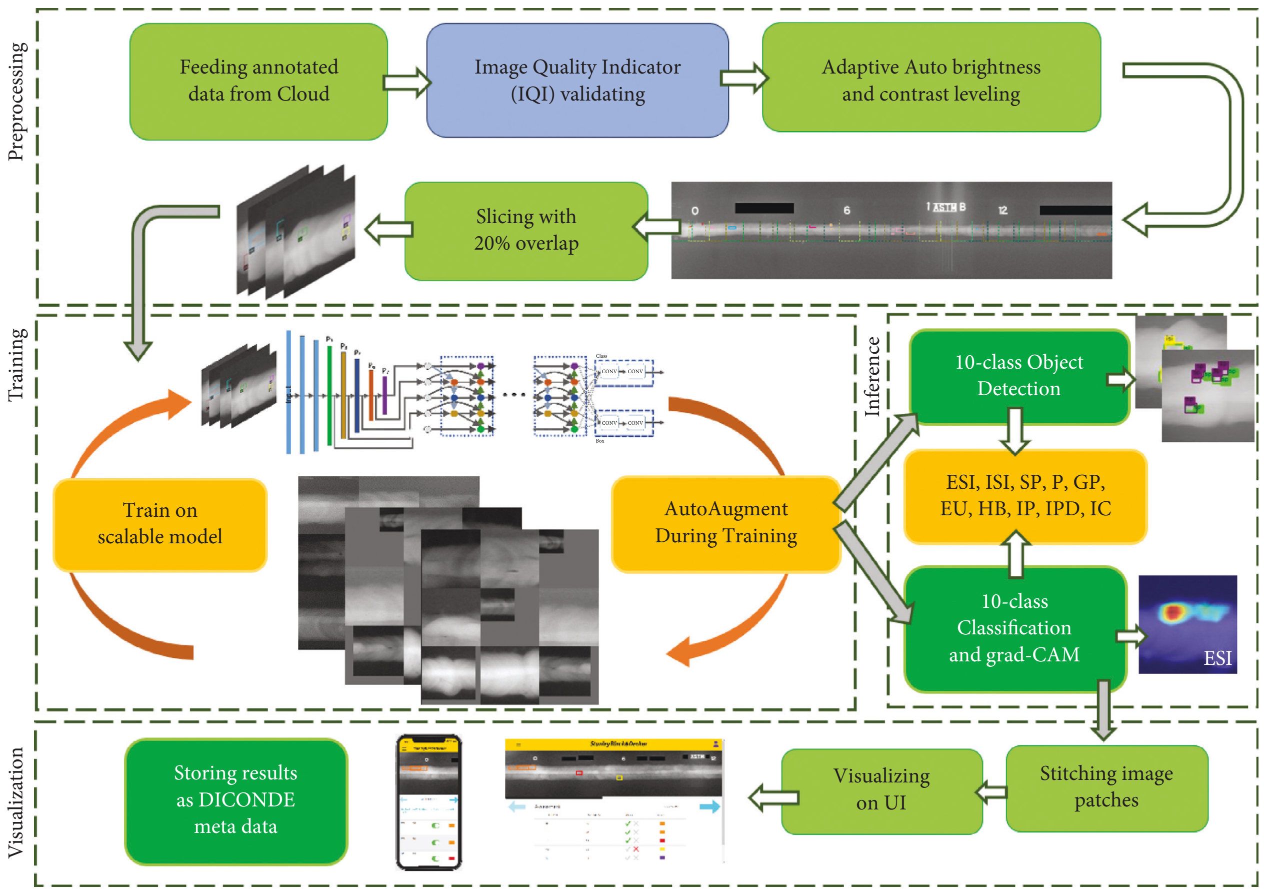

Preprocessing
Feeding annotated data from Cloud
Image Quality Indicator (IQI) validating
Adaptive Auto brightness and contrast leveling
Slicing with 20% overlap
Training
Train on scalable model
AutoAugment During Training
Inference
10-class Object Detection
ESI, ISI, SP, P, GP, EU, HB, IP, IPD, IC
10-class Classification and grad-CAM
ESI
Visualization
Storing results as DICONDE meta data
Visualizing on UI
Stitching image patches

FIGURE 5: An overview of the system architecture: the box in blue accent is optional and can be done once images are captured before the start of the detection process. The arrows in gray color show transitions to the next stage. In section inference, one of the object detection or classification tasks will be done based on the in-use trained model. In visualization, depending on the need, stitching, visualization, storing, or all of them in once can be done.

3.2. Network. As described in Section 2, the final dataset contains 23469 image patches of size 224×244 pixels. An image patch passes through backbone for feature extraction. In this work, EfficientNet [53] is used as the backbone of object detection models and for feature extraction in classification models. However, for weld quality assessment, different backbone performances are evaluated, and class activation maps are reported. Next, multiscale features from levels P_3 to P_7 pass through a successor of feature pyramid networks (FPNs) [57]. P_i denotes the resolution of the input activation map that is $1/2^i$ of the original input image. In conventional FPN, it is assumed that features from various scales contribute equally to the final detection. A few works have investigated the optimization of feature fusion; e.g., NAS-FPN [51] is an effort to find optimum architecture for cross-scale fusing network through search. However, it takes thousands of GPU hours to find an optimal design and the resulting model is oversized. To address the equal contribution of different scales in fusing features, EfficientDet uses bidirectional FPN (BiFPN). In BiFPN, similar to FPN, a top-down pass is used, and similar to PA-Net [58] bottom-up pass is added. Nonetheless, the bottom-up pass adds a lot of costly additional weights to the network. Thus, nodes with single connections (highest and lowest levels) are removed in view of less contribution in feature fusion to optimize the structure. In addition, a few edges from input to output (similar to skip connections in ResNet [36]) are added, which boost both the training and accuracy processes. Finally, fused features pass through two similar class and box networks used to determine the class label and the bounding box location of detected discontinuities. Similar to backbone and BiFPN, depth of class/box nets gets scaled with a single coefficient.

3.3. Scalability. In this part, the single compound scaling coefficient of the overall architecture is reviewed. EfficientDet family starts from the smallest model D0 and ends with the deepest and largest model D7, where the number stands for the single compound coefficient ϕ, used to scale input image resolution and overall depth and width of the architecture. For backbone, if EfficientNet is used, one of the pretrained networks is applied based on ϕ. Figure 6 shows the architecture, which is similar for all networks. The final input image resolution is determined using the following equation:

$$R_{\text{input}} = 512 + \phi \cdot 128. \quad (1)$$

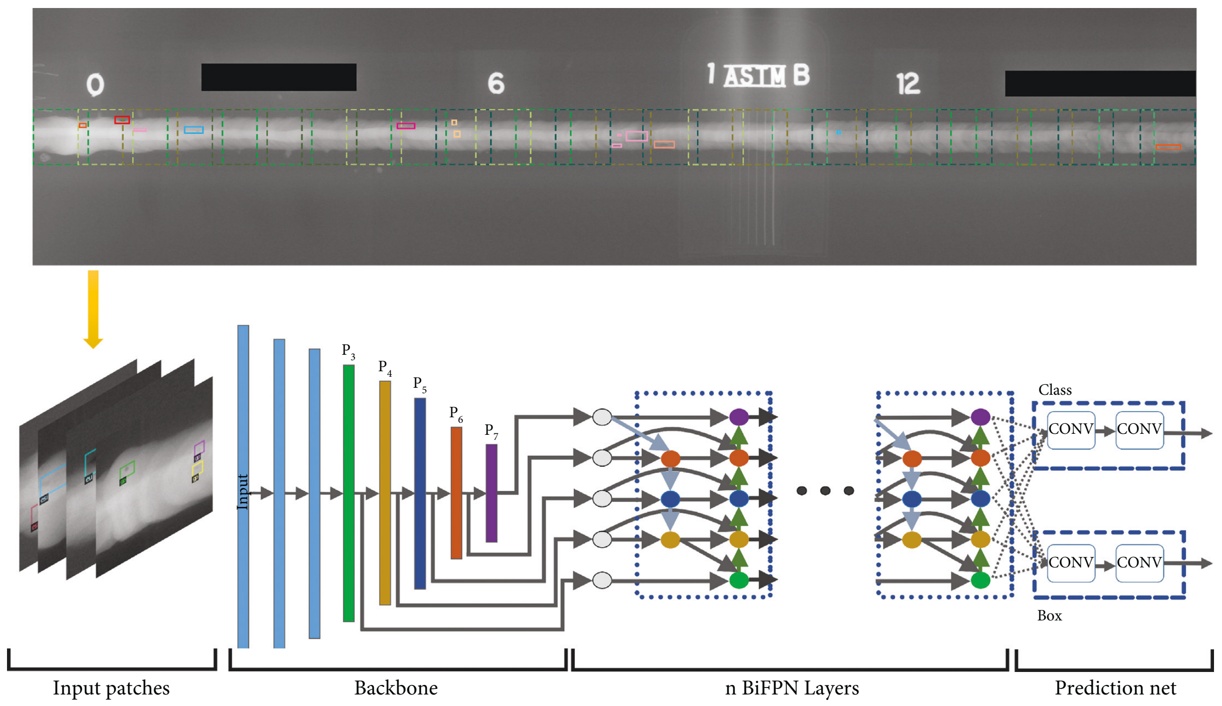


Figure 6: EfficientDet family architecture: images pass through the backbone, and feature scales P3 to P7 get fed into the BiFPN network. Input image resolution is calculated from equation (1). The number of BiFPN repeated blocks extracted using equation (2). Depth of box/class prediction nets is determined using equation (3).

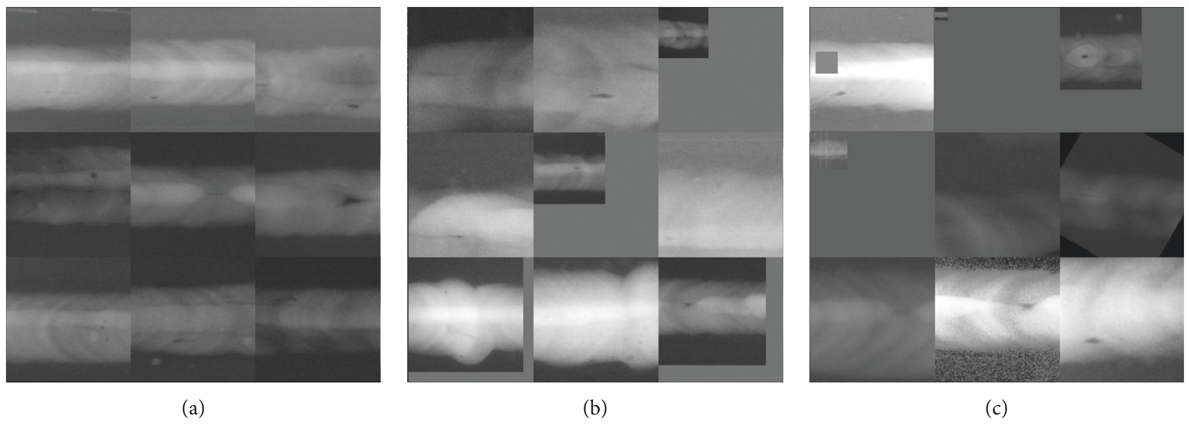


Figure 7: Samples of a training batch with different augmentations: (a) a batch with no augmentation, (b) an augmented batch with a collection of random augmentation named as train-time augmentation, and (c) augmented batch based on policyV3.

Equation (2) shows how the number of channels/layers of the BiFPN is scaled, where each layer is one of the BiFPN repeated blocks starting from 3 for D0 shown in Figure 6. Finally, the number of layers of the class/box is determined through equation (3).

$$\begin{aligned} W_{\text{bifpn}} &= 64 \cdot \left(1.35^{\phi}\right), \\ D_{\text{bifpn}} &= 3 + \phi, \end{aligned} \tag{2}$$

$$D_{\text{box}} = D_{\text{class}} = 3 + \left\lfloor \frac{\phi}{3} \right\rfloor. \tag{3}$$

3.4. Data Augmentation. Many object detection as well as weld quality assessment deep learning approaches employ data augmentation in order to improve both the performance of the network and generalization [31, 40, 43]. The effectiveness of augmentation is shown and evaluated in literature [59]. Nonetheless, there are countless strategies, such as rotation, affine, zoom in/out, flipping, etc., various magnitudes, and also different possible combinations of strategies to be used for augmenting the dataset. One solution is to search through all possible solutions to find the optimal ones. The authors in [60] investigated and searched through the area of 10^{10} different combinations for the classification task. Similarly, [61] investigated the effectiveness of AutoAugmentation for object detection and extracted a few sets of policies enhancing detection performance the best for the object detection task named as policy V0-3. As searching for optimal augmentation strategies is a time-consuming task, extracted policies are applied and investigated in this work. For this purpose, a base model (D0 with EfficientNet B0 backbone) is trained utilizing each of the policies to find the best policy. Then, best policy is used for training larger models and investigating other effective parameters of the model.

3.5. Evaluating Metrics. Evaluating results is performed through average precision (AP) metrics. Models output a bounding box, a corresponding class label, and confidence for each detection. A detection is considered correct when the area of the ground truth bounding box and the detected box have at least 0.5 intersection over the area of the union of two mentioned boxes, which is called Intersection over Union (IoU). Also, the class labels of both bounding boxes should be the same, which means

$$\mathrm{IoU} = \frac{\mathrm{area}\left(\mathrm{B}_{\mathrm{Bp}} \cap \mathrm{B}_{\mathrm{gt}}\right)}{\mathrm{area}\left(\mathrm{B}_{\mathrm{Bp}} \cup \mathrm{B}_{\mathrm{gt}}\right)}. \tag{4}$$

With IoU less than 0.5, the detection is counted as fp. Fn is also the count of nondetected bounding boxes. Therefore, precision and recall are calculated through the following:

$$\text{precision} = \frac{\mathrm{tp}}{\mathrm{tp} + \mathrm{fp}};\ \text{recall} = \frac{\mathrm{tp}}{\mathrm{tp} + \mathrm{fn}}. \tag{5}$$

As recall and precision of a robust object detector do not alter much with varying confidence, it is required to consider multiple confidence thresholds to evaluate the performance of the object detector [62]. Defining all-point interpolation of the area under precision-recall curve obtains accurate results by pruning zig-zag behavior of the curve and utilizing maximum precision ($P_{\mathrm{interp}}\ (r)$ where r is recall level and recall of the point is greater than r_{n+1}) at each recall level, instead of using the precision at that point. The mathematical presentation of this is as follows:

$$\mathrm{AP} = \sum_{n} \left(r_{n+1} - r_n\right) P_{\mathrm{interp}}\left(r_{n+1}\right), \tag{6}$$

where

$$P_{\mathrm{interp}}\left(r_{n+1}\right) = \max_{\tilde{r}:\, \tilde{r} \geq r_{n+1}} P(\tilde{r}). \tag{7}$$

AP has become a standard for comparing model performance in different object detection challenges [63] as well as literature [41, 52, 64, 65].

In Section 4, models are evaluated using mAP (mean AP) (which is equal to mean of AP with IOU threshold ranging from 0.50 to 0.95 and step of 0.05), AP_{50}, AP_{75} (which is equal to ap@iou=0.75), AP_{s} (s stands for small and objects with area < 32^2), AP_{m} (m is medium and area of the objects is between 32^2 and 96^2), and AP_{l} (objects with area > 96^2).

4. Experiments

In this section, various experiments are designed and performed to investigate a set of scalable models with fast processing time while maintaining high accuracy. In addition to EfficientNet backbones, results are reported utilizing other backbones, namely, MobileNetV3 [66], ResNet50 [36], which is called Resdet50 in detection models, CspResdet50 [67], and Darknet (utilized in Yolov3 [54]). Moreover, standalone object detection models including Yolov3, Yolov4 [46], Yolov5 [68], and RetinaNet [65] are fine-tuned as a basis for comparison. In the following sections, the K-means method is used to extract optimal anchor boxes; analysis and results from applying various AutoAugment policies on models, training setup and hyperparameter tuning, quality assessment with single class labels, effects of using several activation functions, and backbones are elaborated, respectively.

4.1. Anchor Boxes. Similar to [57], EfficientDet uses anchor boxes to detect objects. By default, there are three distinct aspect ratios (0.5, 1.0, and 2.0). K-means clustering is utilized to find the set of optimal aspect ratios for the box prediction network [64]. Moreover, the input image resolution is also considered in optimized aspect ratio calculation. Table 2 demonstrates the effectiveness of using aspect ratios calculated by K-means. The results are reported using AP metrics. New aspect ratios (1.2, 2.14, 3.8) suggest that 99.92% (i.e., equivalent to the percentage of bounding boxes that lie into one of K-means calculated clusters) of the defects are horizontal rectangles, and optimizing helps with a 6.6% increase in AP_{50}.

4.2. Analysis of Augmentation Policies. Since training each model employing all policies V0 to V3 is time-consuming, EfficientDet-D0 is considered the base model for analyzing how transferring augmentation policies affect the detection of characteristics. Table 3 demonstrates the effects of utilizing various policies during training for augmentation, based on AP metrics. In NoAugment, raw images are passed to the network, while in Train-timeAug, two common augmentations for train-time are used. First, images are flipped horizontally with a probability of 50%, and second, images are randomly resized and padded with a random scale between 0.1 and 2.0. Also, bilinear interpolation is used while resizing, and the mean of the dataset is applied for padding when the final image is smaller than 512×512 pixels (as 512×512 pixels image is the target image size for the D0 model). PolicyV0-3 refers to each of 4 policies introduced in [61]. In a similar way, during training of the D0 model using each of these policies, a random set of strategies from the selected policy with a probability of 66% is selected, and the input image is augmented based on it (the probability of not performing any of the strategies is one-third). Moreover, similar augmentation is performed on bounding boxes if any is affected. Based on Table 3, augmentation policies dramatically boost the performance of the network by 3.8 to 6.9 AP. Most policies assist the network detect smaller defects

Table 2: Effect of optimizing aspect ratio of anchor boxes based on bounding box annotations in train dataset. For this experiment, EfficientDet-D0 model is used and the optimized aspect ratios are (1.2, 2.14, 3.8).

Default					K-means optimized (% of improvement)				
mAP	AP_{50}	AP_s	AP_m	AP_l	mAP	AP_{50}	AP_s	AP_m	AP_l
30.7	59.3	24.0	33.9	68.3	34.8 (↑**4.1**)	65.9 (↑**6.6**)	25.4 (↑**1.4**)	35.0 (↑**1.1**)	69.2 (↑**0.9**)

(i.e., AP_s which are of more importance since they are easier to miss during manual inspection and are harder to detect with deep learning methods). For further investigations, train-timeAug, and policyV3 are applied to the images as they resulted the best in these experiments. Figure 8 depicts a sample training batch with mentioned augmentations applied.

Table 3: Policies used to train on D0 model on entire train images. Results are reported on the validation set.

Policy name	Base (% of improvement)			
	mAP	AP_s	AP_m	AP_{50}
NoAugment	24.2	21.5	27.2	46.0
Train-timeAug	30.4 (↑**6.2**)	24.4 (↑**2.9**)	39.0 (↑**11.8**)	62.5 (↑**16.5**)
policyV0	28.0 (↑**3.8**)	23.2 (↑**1.7**)	33.0 (↑**5.8**)	56.0 (↑**10.**)
policyV1	30.3 (↑**6.1**)	22.2 (↑**0.7**)	35.1 (↑**7.9**)	61.1 (↑**15.1**)
policyV2	30.2 (↑**6.0**)	22.1 (↑**0.6**)	36.3 (↑**9.1**)	59.7 (↑**13.7**)
policyV3	31.1 (↑**6.9**)	24.9 (↑**3.4**)	37.2 (↑**10.**)	60.9 (↑**14.9**)

4.3. *Training and Hyperparameter Tuning.*

The size of the models and the resolution of input images increase from D0 to D7 gradually using equations (1)–(3). It is not possible to train all the models on GPUs with 16 GB RAM with suitable possible batch size relative to model size. Models with smaller ϕ coefficients (i.e., D0 to D2) are trained on 3 NVIDIA V100 16 GB RAM GPUs with maximum possible batch size though for fitting these models in such GPU memory, a few actions are performed. First, mixed-precision training2 is applied using the Apex package which assists in decreasing memory usage and training time by utilizing half-precision weights and operations if possible. Second, as providing accurate statistics for batch normalization is crucial for the stabilized learning process and high-speed convergence, in distributed training, synchronized batch normalization is used to provide cross-device batch-norm statistics. Nonetheless, these would not help to fit D5 to D7 models in GPU. Thus, results related to those models are not reported. Finally, for comparison, several original Yolo and RetinaNet models are trained. For RetinaNet models, images are resized to 800×800 pixels, and ResNet with 50 or 101 layers are used as the backbone, and for Yolo models, images are resized to 640×640 pixels. Implementation for yolo models can be found in [68], and RetinaNet can be found in detectron2 framework [69].

During training, normalization using precomputed mean and standard deviation values per channel on the entire dataset is performed. Also, each image is first randomly flipped horizontally and/or resized for all experiments (i.e., Train-timeAug explained in Section 3.4). For weight initialization, the weights originally were trained on MS COCO dataset in [52] and are converted to PyTorch in [70]. Thus, all weights of the network are trained to reach maximum performance similar to our previous work [56].

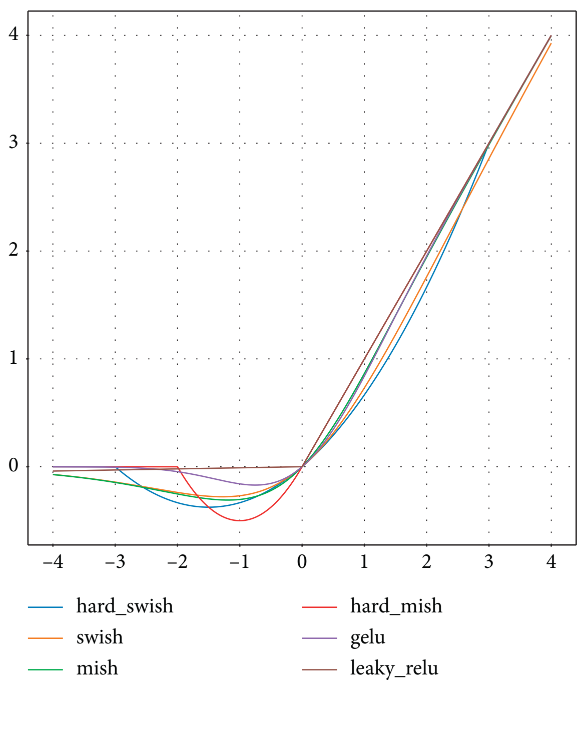
Identical to [52], cosine learning [71] is used. At the beginning of the training process, for the first few epochs (epoch numbers 0 to 5) learning rate increases gradually to the desired point, and from epoch 5 to the end of the training process the learning rate decreases gradually in cosine form. In addition, learning rate noises applied to 30% and 90% of the training process. Moreover, in a few experiments, exponential moving average (EMA) [72] with a weight decay of 0.9998 was applied; however, it was removed as non-EMA training ended up with higher AP. Furthermore, a few optimizers are evaluated and results show in this task Fusedadam [73] optimizer converges faster and reaches a higher accuracy (0.6 mAP). In the following, the impact of different activation functions is discussed.

4.4. *Effect of Different Activation Functions.*

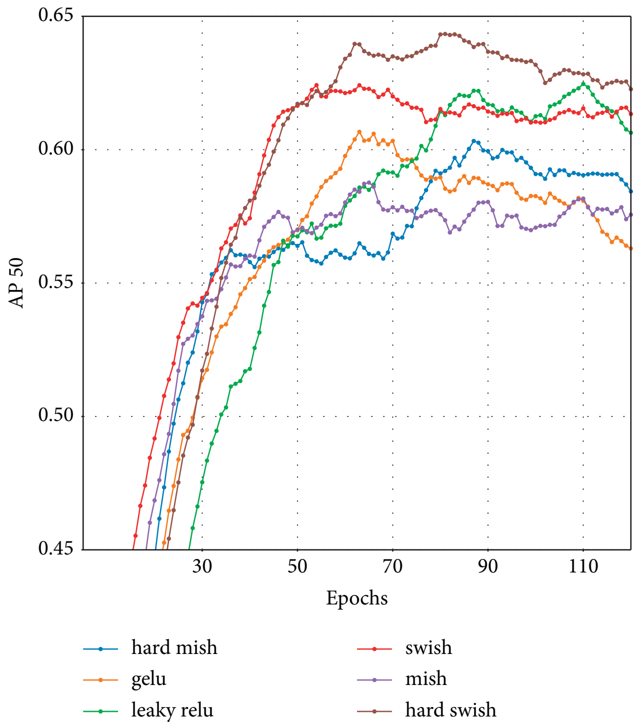
The performance of the base model (i.e., EfficientDet-D0) is analyzed by testing over different activation functions, namely, Leaky Rectified Linear Unit (Leaky ReLU), Gaussian Error Linear Units (GeLU) [74], Swish [75], Mish [76], and hard Swish [66] in which sigmoid is replaced with relu6$(x+3)/6$, which is more memory efficient, and hard Mish [77]. Figure 9(a) visualizes mentioned activation functions. Note that the specified activation function is used for BiFPN layers and class/box prediction nets. As shown in Figure 9(b), both hard Swish and Swish outperforms other activation functions based on AP_{50}. The same improvement applies for other AP metrics. However, this did not happen for deeper models of the EfficientDet family. Thus, default Swish is used to maximize model performance, though in view of the memory efficiency of hard Swish, it is a preferable choice for activation function if the model is planned to get deployed on hardware-constraint end-device. For non-EfficientDet family models, the default activation function of the model is used.

4.5. *Defect Object Detection without considering Class Labels.*

In this experiment, all discontinuities are considered with a single *defect* label. Table 4 shows network performance considering a single class for all discontinuities. Although these models only perform localization of the defects and no class label is available, higher accuracy in localization is reached. In addition to EfficientDet Family, several other models are also added for sake of comparison. All models are



(a)



(b)

Figure 8: Activation functions used for comparing performance of the models.

trained using base parameters in their original work, except for Yolov3, in which focal loss is preferred to address class imbalance. In Section 4.6, models in Table 4 are discussed.

4.6. Evaluating Results. Tables 4–6 demonstrate models evaluation on 10-class object detection task, 10-class classification, and defect/nondefect object detection task, respectively. All results for object detection models are obtained from the test set. For all experiments, common train-time augmentations and policyV3 (described in Section 3.4) are applied. Although hard Swish improved accuracy for shallower models, the same did not happen for deeper ones (D3 and deeper). Thus, default activation functions of the models are used (i.e., Swish for EfficientDet models, leaky-ReLU for CspResdet50 and darkdet53, and ReLU for Resdet50). As mentioned in the tables, generally, deeper models show more accurate performance. However, little improvement or minor deterioration in larger models is a result of having to use smaller batch size to be able to fit the model into the GPUs (i.e., batch size of 20 per GPU is used to train the D0 versus batch sizes 3 and 1 for D3 and D4 models, respectively), which accounts for inaccurate estimation of statistics of batch normalization and deteriorates training process. Since for task of weld quality assessment and indexing of weld as well as rejecting or accepting, predicting 50% of the discontinuity is acceptable, AP_{50} is used for further model comparison and analysis. AP_l is not reported because defects in welds with area greater than 96^2 pixels are undersampled and uncommon. Therefore, it would not be a reliable measure to evaluate the performance of the models, and it is not reported.

For inference time analysis, a similar GPU that is used for training, NVIDIA V100 16 GB, is exploited. Inference times in Tables 4 and 6 suggest that models are able to perform in real-time performance based on definition of real-time for object detection models [78]. As a result, the fastest and the most accurate models can infer up to 224 and 150 image patches per second with a batch size of 16, respectively. Considering the fact that in the worst case each complete weld image has a length of 15360 pixels and is cropped with 20% overlap, a full image will have about 86 patches, meaning models can infer an image in 385 to 465 ms. Thus, models are able to process weld images in real time with consideration of required preprocessing. Figure 10 summarizes models' latency and floating-point operations (FLOPs) count. Yolo models have a higher number of operations, and the resulting models are larger. In contrast, the EfficientDet family models and models with Bi-FPN Layers enhanced with AutoAugmentation are both smaller and more accurate. Although EfficientDet models have a smaller number of parameters, they perform slower on GPU because of slower execution of separable convolution. Finally, a fusion of Resnet50 with EfficientDet object detection architecture results in best accuracy versus latency, for this task.

In Tables 4 and 6, models reported above double line are trained for the sake of comparison. In Yolo models in addition to Train-timeAug, mosaic augmentation is applied. Although this improved results by 0.5 AP, EfficientDet

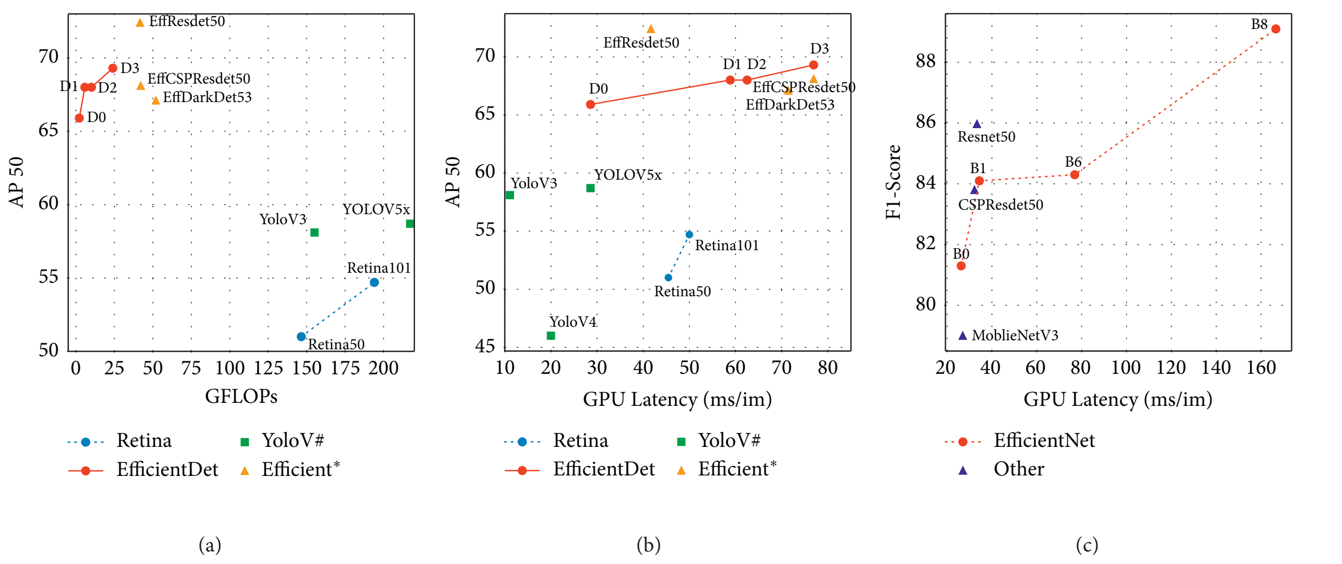


Figure 9: (a) AP_{50} accuracy versus giga floating-point operations (GFLOPs) for 10-class object detection models. (b) AP_{50} versus GPU latency (milliseconds/224 × 224 image patch) for 10-class object detection. (c) F1-score versus GPU latency (ms/224 × 224 image patch) performance for 10-class classification models.

Table 4: Model defect detection performance on test set based on defect/nondefect labels.

Model Name	# of params	Test					Inference time (images/sec)		
		AP	AP_{50}	AP_{75}	AP_s	AP_m	$b = 1$	$b = 8$	$b = 16$
Yolo v3	61.4 M	34.3	81.0	—	—	—	54	65	75
Yolo v4	64.3 M	27.4	61.6	—	—	—	48	130	185
Yolo v5x	87 M	34.8	81.9	—	—	—	35	50	56
RetinaNet-50	37.9 M	29.6	73.6	14.5	29.5	30.5	22	—	—
RetinaNet-101	56.9 M	30.9	75.2	14.2	29.9	31.5	20	—	—
EfficientDet-D0	3.8 M	37.1	87.5	18.5	37.5	37.3	35	130	185
EfficientDet-D3	11.9 M	37.7	88.1	19.7	36.7	38.8	14	56	65
Efficientdarkdet53	11.9 M	36.6	85.7	18.9	38.4	35.5	15	34	37
EfficientCspResDet50	23.6 M	37.6	86.2	20.2	38.8	36.8	15	36	39
EfficientResDet50	**26.9 M**	**38.2**	**89.0**	**21.8**	**37.8**	**40.9**	**27**	**143**	**168**

Inference times are reported based on tests on a V100 GPU card. For EfficientDet models, number D# stands for ϕ coefficient of the model, and depth and input image resolution can be acquired using equations (1)–(3). Also, the number stands for backbone number in [53]. For other models, the same structure of EfficientDet-D1 is used except the backbone, the name of which is determined in Model Name. For models without efficient in their name, implementations from [69] or [68] with common augmentation are employed.

Table 5: 10-class classification backbone performances.

Model Name	Image Size	# of params	Top-1 accuracy	Precision	Recall	F1-score	Inference time (images/sec)	
							$b = 1$	$b = 16$
Resnet50	224	23.5 M	86.83	85.68	86.26	85.97	30	207
CspResnet50	224	20.6 M	85.3	83.89	83.71	83.8	31	266
MobileNetV3	224	4.2 M	79.91	79.41	78.67	79.04	37	267
EfficientNet-b0	224	3.8 M	82.15	81.4	81.36	81.38	38	355
EfficientNet-b1	240	6.5 M	85.4	83.8	84.4	84.1	29	200
EfficientNet-b6	528	40.7 M	85.71	84.5	84.1	84.3	13	26
EfficientNet-b8	**672**	**89.6 M**	**90.2**	**89.5**	**88.72**	**89.11**	**6**	**10**

models are still more accurate. In contrast to large number of parameters in Yolo, they still perform faster than EfficientDet models and the reason is that depth-wise separable convolutions is employed for feature fusion in EfficientDet and they run slower on GPU. However, thanks to lower number parameters, these models will perform better on

Table 6: 10-class defect detection models performance on test set based and inference time with multiple batch sizes.

Model Name	# of params (M)	Test set					Inference time		
		AP	AP_{50}	AP_{75}	AP_s	P_m	b ${}^{b}=1$	$b=8$	$b=16$
YOLO V3	61.5 M	30.9	58.4	—	—	—	0	150	150
YOLO V4	64.3 M	24.1	46.0	—	—	—	0	120	125
YOLO V5x	87.2 M	30.1	58.1	—	—	—	5	50	56
RetinaNet50	37.9 M	26.7	51.0	18.4	20.	27.6	21	—	—
RetinaNet101	56.9 M	27.7	54.7	18.2	21.	30.7	20	—	—
EfficientDet-D0	3.8 M	34.8	65.9	22.2	2.4	35.0	35	144	224
EfficientDet-D1	6.5 M	34.9	68.0	23.7	27.3	34.4	17	111	148
EfficientDet-D2	8M	35.2	68.0	23.0	26.0	36.5	16	88	106
EfficientDet-D3	11.9 M	34.7	69.3	25.1	25.9	34.9	13	58	66
Efficientdarkdet53	11.9 M	34.4	67.1	25.4	264	33.9	14	33	37
EffcientCspResdet50	23.7 M	34.1	68.1	22.5	25.7	33.6	13	35	39
EfficientResDet50	**27M**	**36.1**	**72.4**	**25.0**	**26.0**	**37.7**	**24**	**132**	**150**

For EfficientDet models, number D# stands for ϕ coefficient of the model and depth and input image resolution can be acquired using equations (1)–(3). Also, the number stands for backbone number in [53]. For other models, the same structure of EfficientDet-D1 is used except the backbone. Note that all models are optimized for maximum AP. b denotes batch size for inference.

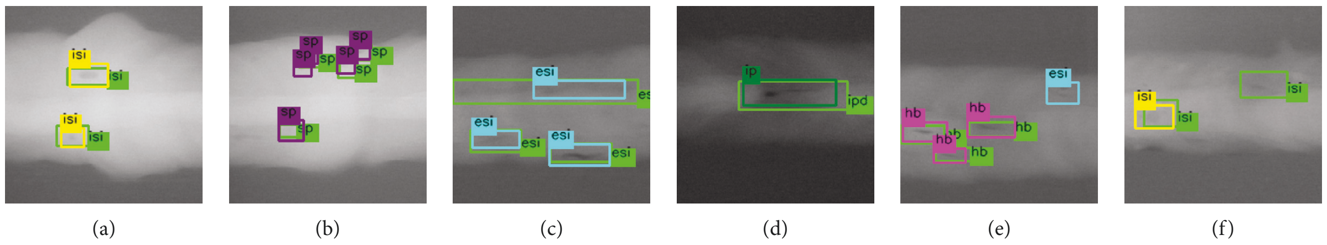


Figure 10: Examples of network prediction: true positives (a, b) (green boxes denote ground truth and others are network prediction), false negatives and false positive as a result of IoU < 0.5 (c), erroneous class prediction because of the similarity of the classes IP and IPD (d), and hard samples which resulted in incorrect prediction as a consequence of not meeting minimum standards (e, f).

CPUs compared with competitors. Results from Table 4, in which for all discontinuities a single label is used, suggest that a portion of false-positive detections are related to detecting correct labels. In the following, erroneous and missed detections of the best model are elaborated upon. Also, the performance of feature extraction backbones both numerically and visually is discussed.

Error analysis: based on Table 6 and inferred images of the test set, erroneous detections of the network with highest AP_{50} belong to one of these subcategories: (1) errors as a consequence of inadequate IoU of detections and ground truth: 0.7% of error cases of test set belong to this category, and they are from 3 classes of IP, ESI, and GP, where 76% of cases are ESI. Figure 11(c) shows a sample from this category. (2) False positives were mostly related to instances that a nondefect bounding box is detected, and it is closely similar to one of the other defect classes, and a nonexpert observer might consider it as a defect. However, it does not meet minimum requirements such as length for slag inclusion, size for porosity, and other criteria to be counted as a discontinuity. Finally, out of 4.5% of instances lying in this category, slag inclusions and HBs had the largest normalized percentage of errors. Mostly, sides of the weld root and also weld toe were falsely predicted as slag inclusions. Figure 11(e) is a sample of this category. As a workaround to reduce the error rate of this type, adding a large number of similar image patches to the train set is suggested. (3) False negatives are where the network does not detect defects. With more than 12% of false negatives, this group contains the largest erroneous behaviour of the model, with HBs and ESIs forming more than 55% of the normalized number of false positive detections. Figure 11(f) is a sample of a false negative. A suggested workaround is to perform online or offline hard example mining for training. Note that by lowering the minimum confidence threshold, most of these are detected concisely by the network. (4) Misclassified samples are when the network detects the object with acceptable IoU, though the class label is incorrect. A sample is shown in Figure 11(d). Finally, Figure 12(a) shows the distribution of misclassified samples from the 10-class object detection model. It is showing that HB class has the most misclassified detections, and it is mostly mistaken with class IC, and the similarity is that both IC and HB create a hallow area in the weld root.

Comparison of backbones: although it is common that multiple discontinuity types appear in a single patch, a part of the dataset (which includes around 80% of each set) that holds image patches with a single defect type is separated and used to evaluate feature extraction and backbone performance, and also to train a classification model. Table 5 shows performance of various backbones. The most accurate backbone is EfficientNet-B8, with 90.2% accuracy on the validation set. A similar training environment and optimizer with object

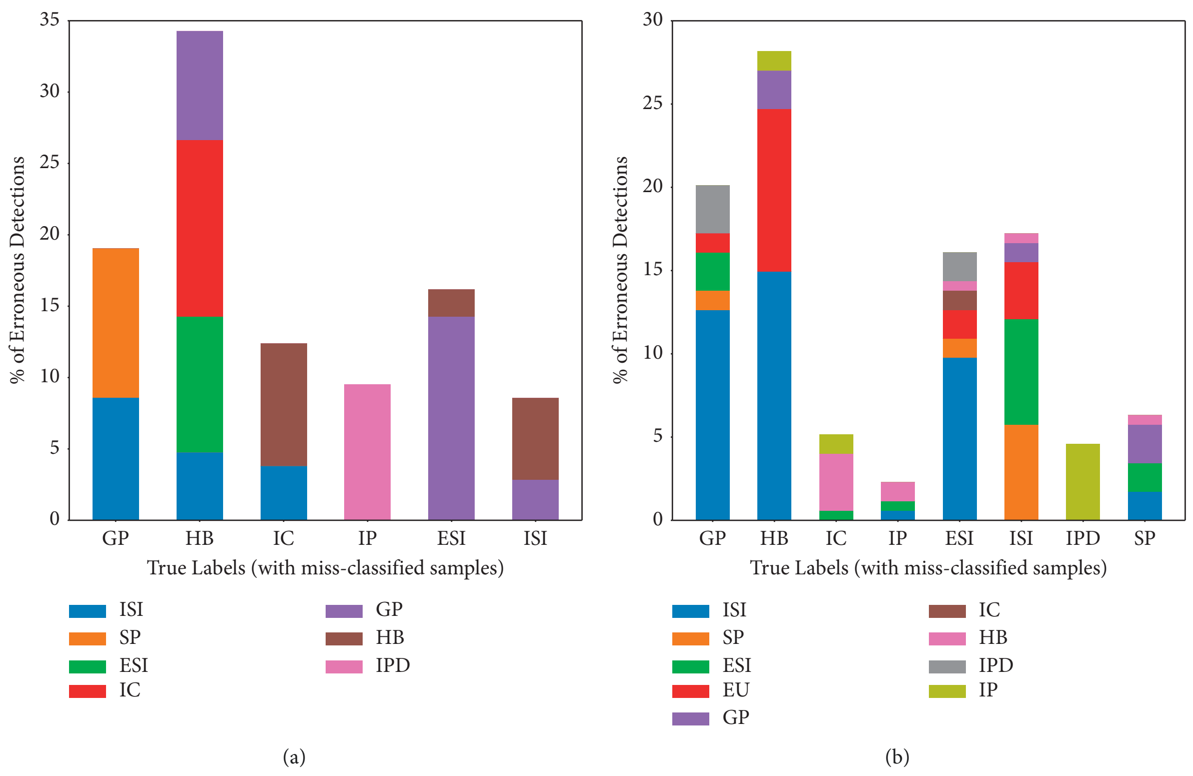


Figure 11: Misclassified detected defects. (a) Misclassification distribution for 10-class object detection model and (b) number of misclassified patches in classification backbone (EfficientNet-b8).

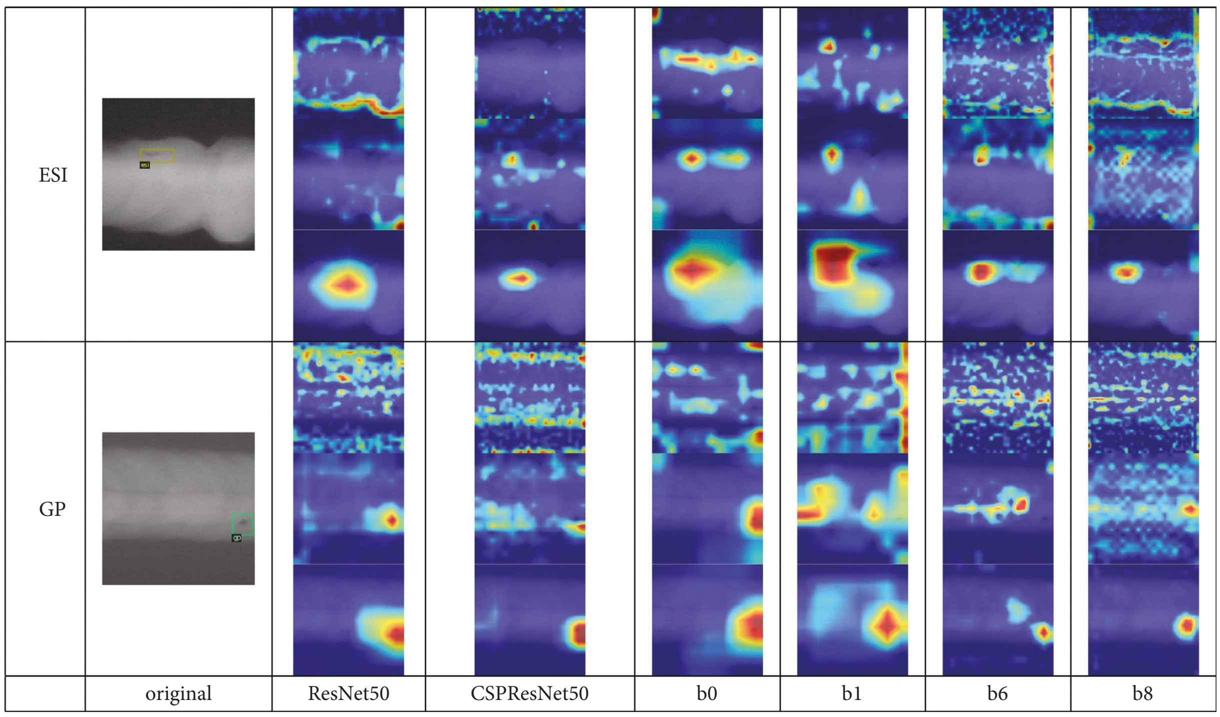


Figure 12: Grad-CAM visualization of different backbones from their three last blocks for classes GP and ESI.

detection models are used for this purpose. Transfer learning is applied and weights were originally trained on ImageNet [77]. Figure 12(b) shows the distribution of erroneous behavior of the classifier. Based on Figure 12(b), most of the misclassified samples are related to HB. Also, erroneous detections are mostly misclassified as ISI, as shown in Figure 12(b).

Explainability using Grad-CAM: gradient-weighted class activation mappings (Grad-CAMs) [79] recognizes the parts of input image with deterministic role in final decision-making of the model. In Grad-CAM, instead of applying global average pooling as ending layers [80] which requires model modification and affects the network performance, back-propagation is utilized to extract feature contributions. Therefore, class activation maps get extracted precisely. In Figure 7, Grad-CAMs of various backbones with different depth are visualized, which provides local explainability for input images. Bottom image of each cell shows final layer Grad-CAM, and upper images are second to last and third from last block output. It shows how network gradually attend to discriminative features of each image.

5. Conclusions

In this paper, a scalable and efficient family of deep models for 10-class weld quality assessment using object detection is presented. A comparative analysis on various models is also performed; several critical elements of the networks such as activation functions and hyperparameters are explored and tuned to achieve state-of-the-art results on the dataset. Moreover, the effects of transferring object detection AutoAugment policies are surveyed. Furthermore, various scenarios such as considering task as a classification only task and defect/nondefect scenarios are also analyzed and models are compared with main-stream object detection models in real-time applications. Finally, model visual explainability is analyzed through employing Grad-CAM and visualizing gradient information for target class. The results are interpreted. They demonstrate that models are able to infer a complete welded joint (15360×1024 resolution X-ray Image) in 385 milliseconds. Although classification task outperforms object detection models, localization of the defect (whether the defect is on root pass, fill pass, or cover pass) is necessary for further indexing of the weld, pass or rejection, and optimization of welding operation.

Traditional computer vision techniques for weld defect detection require several critical preprocessing steps resulting in a nonrobust outcome or human intervention is needed. In contrast, automatic feature extraction approaches and deep learning-based methods require minimum human intervention or preprocessing to achieve state-of-the-art results. The models presented here can be used as assistive defect-recognition systems to facilitate robust defect localization and classification and to reduce both human workload and error. Finally, as experts may have conflicting and personal performance in particular defect detection, provided deep models may train on specific samples and predict defects with a consolidated standard which can also be helpful in training experts.

Future works contain test-time augmentation, model ensemble without sacrificing real-time capability of the system, searching for optimal auto augmentation policies utilizing reinforcement learning since policies were initially extracted from the COCO dataset and the nature of the weld images is not consistent with nature of COCO dataset images. In addition, through time, more samples will be gathered from various sites of different parts of the world, and the dataset will expand in both the number of classes and the number of instances per class.

Data Availability

All open-source implementations used in this paper are referenced in the main body of the article. However, the remaining implementations and dataset are a part of on-going research and proprietary of Stanley Black & Decker, USA.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this study.

Acknowledgments

This work was done primarily with the help and support from Michael George, Jeremy Guretzki, Matthew Nelson, Jason Miller, William Aston, Jake Smith, Haresh Ghansyam, Pete Morris, Prashanth Tirumalaseti, Adam Wynne Hughes, and Shengnan Wang as well as great support from Dr. Mark Maybury and Dr. Manish Mehta from the office of CTO at Stanley Black and Decker. This research was conducted at Lamar University and was fully funded by Artificial Intelligence Lab, Stanley Oil & Gas, Stanley Black & Decker, USA.

References

[1] R. Vilara, "An automatic system of classification of weld defects in radiographic images," *NDT & E International*, vol. 42, no. 5, pp. 467–476, 2009.

[2] M.-M. Naddaf-Sh, S. Naddaf-Sh, H. Zargaradeh et al., "Next-generation of weld quality assessment using deep learning and digital radiography," in *Proceedings of the AAAI Spring Symposium Series*, Palo Alto, CA, USA, March 2020.

[3] M.-M. Naddaf-Sh, S. Naddaf-Sh, H. Zargarzadeh et al., "Defect detection and classification in welding using deep learning and digital radiography," in *Fault Diagnosis and Prognosis Techniques for Complex Engineering Systems*vol. 2021, pp. 327–352, 2021.

[4] A. Shahriar, R. Sadiq, and S. Tesfamariam, "Risk analysis for oil & gas pipelines: a sustainability assessment approach using fuzzy based bow-tie analysis," *Journal of Loss Prevention in the Process Industries*, vol. 25, no. 3, pp. 505–523, 2012.

[5] Z. Rui, G. Han, H. Zhang, S. Wang, H. Pu, and K. Ling, "A new model to evaluate two leak points in a gas pipeline," *Journal of Natural Gas Science and Engineering*, vol. 46, pp. 491–497, 2017.

[6] L. T. Popoola, A. S. Grema, G. K. Latinwo, B. Gutti, and A. S. Balogun, "Corrosion problems during oil and gas production and its mitigation," *International Journal of Integrated Care*, vol. 4, no. 1, pp. 1–15, 2013.

[7] S. M. Anouncia and R. Saravanan, "Non-destructive testing using radiographic images? a survey," *Insight-Non-Destructive Testing and Condition Monitoring*, vol. 48, no. 10, pp. 592–597, 2006.

[8] X. Dong, C. J. Taylor, and T. F. Cootes, "Small defect detection using convolutional neural network features and random forests," *Lecture Notes in Computer Science*, vol. 11132, pp. 398–412, 2019.

[9] A. Shukla and H. Karki, "Application of robotics in offshore oil and gas industry-a review part II," *Robotics and Autonomous Systems*, vol. 75, pp. 508–524, 2016.

[10] L. Yang, Y. Liu, and J. Peng, "Advances techniques of the structured light sensing in intelligent welding robots: a review," *International Journal of Advanced Manufacturing Technology*, vol. 110, pp. 1–20, 2020.

[11] S. Habibian, M. Dadvar, B. Peykari et al., "Design and implementation of a maxi-sized mobile robot (karo) for rescue missions," *ROBOMECH Journal*, vol. 8, 2021.

[12] M. Dadvar and S. Habibian, "Contemporary research trends in response robotics," 2021, https://arxiv.org/abs/2105.07812.

[13] Q. Ma, G. Tian, Y. Zeng et al., "Pipeline in-line inspection method, instrumentation and data management," *Sensors*, vol. 21, no. 11, 2021.

[14] W. Hou, Y. Wei, J. Guo, Y. Jin, and C. Zhu, "Automatic detection of welding defects using deep neural network," *Journal of Physics: Conference Series*, vol. 933, no. 1, 2018.

[15] X. Dong, C. J. Taylor, and T. F. Cootes, "A random forest-based automatic inspection system for aerospace welds in x-ray images," *IEEE Transactions on Automation Science and Engineering*, pp. 1–14, 2020.

[16] D. Mery and M. A. Berti, "Automatic detection of welding defects using texture features," *Insight-Non-Destructive Testing and Condition Monitoring*, vol. 45, no. 10, pp. 676–681, 2003.

[17] J. Kumar, R. Anand, and S. Srivastava, "Multi-class welding flaws classification using texture feature for radiographic images," in *Proceedings of the International Conference on Advances in Electrical Engineering (ICAEE)*, pp. 1–4, Vellore, India, 2014.

[18] J. Kumar, R. S. Anand, and S. P. Srivastava, "Flaws classification using ann for radiographic weld images," in *Proceedings of the 2014 International Conference on Signal Processing and Integrated Networks (SPIN)*, pp. 145–150, Noida, India, February 2014.

[19] J. Hassan, A. M. Awan, and A. Jalil, "Welding defect detection and classification using geometric features," in *Proceedings of the 2012 10th International Conference on Frontiers of Information Technology*, pp. 139–144, Islamabad, Pakistan, December 2012.

[20] O. Zahran, H. Kasban, M. El-Kordy, and F. E. A. El-Samie, "Automatic weld defect identification from radiographic images," *NDT & E International*, vol. 57, pp. 26–35, 2013.

[21] T. Y. Lim, M. M. Ratnam, and M. A. Khalid, "Automatic classification of weld defects using simulated data and an mlp neural network," *Insight-Non-Destructive Testing and Condition Monitoring*, vol. 49, no. 3, pp. 154–159, 2007.

[22] J. Zapata, R. Vilar, and R. Ruiz, "Performance evaluation of an automatic inspection system of weld defects in radiographic images based on neuro-classifiers," *Expert Systems with Applications*, vol. 38, no. 7, pp. 8812–8824, 2011.

[23] I. Valavanis and D. Kosmopoulos, "Multiclass defect detection and classification in weld radiographic images using geometric and texture features," *Expert Systems with Applications*, vol. 37, no. 12, pp. 7606–7614, 2010.

[24] W. Hou, D. Zhang, Y. Wei, J. Guo, and X. Zhang, "Review on computer aided weld defect detection from radiography images," *Applied Sciences*, vol. 10, no. 5, p. 1878, 2020.

[25] M. Carrasco and D. Mery, "Segmentation of welding defects using a robust algorithm," *Materials Evaluation*, vol. 62, no. 11, pp. 1142–1147, 2004.

[26] D. Mery, "Automated detection of welding discontinuities without segmentation," *Materials Evaluation*, vol. 69, no. 6, pp. 656–663, 2011.

[27] M. Ben Gharsallah and E. Ben Braiek, "Weld inspection based on radiography image segmentation with level set active contour guided off-center saliency map," *Advances in Materials Science and Engineering*, vol. 2015, Article ID 871602, 10 pages, 2015.

[28] C. Ajmi, S. E. Ferchichi, and K. Laabidi, "New procedure for weld defect detection based-gabor filter," in *Proceedings of the 2018 International Conference on Advanced Systems and Electric Technologies (ICASET)*, pp. 11–16, Hammamet, Tunisia, March 2018.

[29] O. Ronneberger, P. Fischer, and T. Brox, "U-net: convolutional networks for biomedical image segmentation," in *Proceedings of the International Conference on Medical image computing and computer-assisted intervention*, pp. 234–241, Munich, Germany, October 2015.

[30] W. Hou, Y. Wei, Y. Jin, and C. Zhu, "Deep features based on a dcnn model for classifying imbalanced weld flaw types," *Measurement*, vol. 131, pp. 482–489, 2019.

[31] C. Ajmi, J. Zapata, S. Elferchichi, A. Zaafouri, and K. Laabidi, "Deep learning technology for weld defects classification based on transfer learning and activation features," *Advances in Materials Science and Engineering*, vol. 2020, Article ID 1574350, 16 pages, 2020.

[32] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in Neural Information Processing Systems*, vol. 25, pp. 1097–1105, 2012.

[33] D. Mery, V. Riffo, U. Zscherpel et al., "GDXray: the database of X-ray images for nondestructive testing," *Journal of Nondestructive Evaluation*, vol. 34, no. 4, pp. 1–12, 2015.

[34] R. Miao, Z. Jiang, Q. Zhou et al., "Online inspection of narrow overlap weld quality using two-stage convolution neural network image recognition," *Machine Vision and Applications*, vol. 32, no. 1, pp. 1–14, 2021.

[35] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2015, https://arxiv.org/abs/1409.1556.

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

[37] Q. Wang, W. Jiao, P. Wang, and Y. Zhang, "A tutorial on deep learning-based data analytics in manufacturing through a welding case study," *Journal of Manufacturing Processes*, vol. 63, pp. 2–13, 2021.

[38] A. Paszke, S. Gross, F. Massa et al., "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d' Alché-Buc, E. Fox, and R. Garnett, Eds., Curran Associates, Inc., Red Hook, NJ, USA, 2019.

[39] D. Mery and C. Arteta, "Automatic defect recognition in x-ray testing using computer vision," in *Proceedings of the 2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 1026–1035, Santa Rosa, CA, USA, March 2017.

[40] S. J. Oh, M. J. Jung, C. Lim, and S. C. Shin, "Automatic detection of welding defects using faster R-CNN," *Applied Sciences*, vol. 10, no. 23, pp. 1–10, 2020.

[41] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: towards real-time object detection with region proposal networks," vol. 28, pp. 91–99, 2015.

[42] R. Guo, H. Liu, G. Xie, and Y. Zhang, "Weld defect detection from imbalanced radiographic images based on contrast enhancement conditional generative adversarial network and transfer learning," *IEEE Sensors Journal*, 2021.

[43] L. Yang, H. Wang, B. Huo, F. Li, and Y. Liu, "An automatic welding defect location algorithm based on deep learning," *NDT & E International*, vol. 120, Article ID 102435, 2021.

[44] M. Mirza and S. Osindero, "Conditional generative adversarial nets," 2014, https://arxiv.org/abs/1411.1784.

[45] T. Gantala and K. Balasubramaniam, "Automated defect recognition for welds using simulation assisted tfm imaging with artificial intelligence," *Journal of Nondestructive Evaluation*, vol. 40, no. 1, pp. 1–24, 2021.

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

[47] Y. Yan, D. Liu, B. Gao, G. Y. Tian, and Z. C. Cai, "A deep learning-based ultrasonic pattern recognition method for inspecting girth weld cracking of gas pipeline," *IEEE Sensors Journal*, vol. 20, no. 14, pp. 7997–8006, 2020.

[48] J. Hu, W. Xu, B. Gao et al., "Pattern deep region learning for crack detection in thermography diagnosis system," *Metals*, vol. 8, no. 8, 2018.

[49] American Petroleum Institute, *API 5L: Specification for Line Pipe*, American Petroleum Institute, Washington, NJ, USA, 2004.

[50] American Petroleum Institute, *API 1104: Standard for Welding of Pipelines and Related Facilities*, American Petroleum Institute, Washington, NJ, USA, 2001.

[51] G. Ghiasi, T.-Y. Lin, and Q. V. Le, "Nas-fpn: learning scalable feature pyramid architecture for object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7036–7045, Long Beach, CA, USA, June 2019.

[52] M. Tan, R. Pang, and Q. V. Le, "Efficientdet: scalable and efficient object detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10781–10790, Seattle, WA, USA, June 2020.

[53] M. Tan and Q. Le, "Efficientnet: rethinking model scaling for convolutional neural networks," in *Proceedings of the International Conference on Machine Learning*, pp. 6105–6114, Beach, CA, USA, June 2019.

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

[55] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE International Conference on computer Vision*, pp. 1440–1448, Santiago, Chile, December 2015.

[56] S. Naddaf-Sh, M.-M. Naddaf-Sh, A. R. Kashani, and H. Zargarzadeh, "An efficient and scalable deep learning approach for road damage detection," in *Proceedings of the 2020 IEEE International Conference on Big Data (Big Data)*, pp. 5602–5608, Atlanta, GA, USA, December 2020.

[57] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2117–2125, Honolulu, HI, USA, July 2017.

[58] S. Liu, L. Qi, H. Qin, J. Shi, and J. Jia, "Path aggregation network for instance segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 8759–8768, Salt Lake City, UT, USA, June 2018.

[59] L. Perez and J. Wang, "The effectiveness of data augmentation in image classification using deep learning," 2017, https://arxiv.org/abs/1712.04621.

[60] E. D. Cubuk, B. Zoph, D. Mane, V. Vasudevan, and Q. V. Le, "Autoaugment: Learning augmentation strategies from data," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 113–123, Long Beach, CA, USA, June 2019.

[61] B. Zoph, E. D. Cubuk, G. Ghiasi, T.-Y. Lin, J. Shlens, and Q. V. Le, "Learning data augmentation strategies for object detection," in *Proceedings of the European Conference on Computer Vision*, pp. 566–583, Glasgow, UK, August 2020.

[62] R. Padilla, S. L. Netto, and E. A. B. da Silva, "A survey on performance metrics for object-detection algorithms," in *Proceedings of the 2020 International Conference on Systems, Signals and Image Processing*, pp. 237–242, 2020.

[63] "Coco detection challenge (bounding box)," 2021, https://cocodataset.org/#detection-eval.

[64] J. Redmon and A. Farhadi, "Yolo9000: better, faster, stronger," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7263–7271, Honolulu, HI, USA, July 2017.

[65] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2980–2988, Venice, Italy, October 2017.

[66] A. Howard, M. Sandler, G. Chu et al., "Searching for Mobilenetv3," 2019, https://arxiv.org/abs/1905.02244.

[67] C.-Y. Wang, H.-Y. M. Liao, I.-H. Yeh, Y.-H. Wu, P.-Y. Chen, and J.-W. Hsieh, "Cspnet: a new backbone that can enhance learning capability of Cnn," 2019, https://arxiv.org/abs/1911.11929.

[68] G. Jocher, A. Stoken, J. Borovec et al., "Ultralytics/yolov5: v5.0-YOLOv5-P6 1280 models, AWS, Supervise.Ly and YouTube integrations," 2021, https://zenodo.org/record/4679653#.YTmqdrAzbIU.

[69] Y. Wu, A. Kirillov, F. Massa, W.-Y. Lo, and R. Girshick, "Detectron2," 2019, https://github.com/facebookresearch/detectron2.

[70] 2020 Efficientdet (A Pytorch Implementation of Efficientdet).

[71] I. Loshchilov and F. Hutter, "Sgdr: stochastic gradient descent with warm restarts," 2016, https://arxiv.org/abs/1608.03983.

[72] 2021 Exponential Moving Average." https://www.tensorflow.org/api_docs/python/tf/train/ExponentialMovingAverage.

[73] 2021 Nvidia Apex Optimizers." https://nvidia.github.io/apex/optimizers.html.

[74] D. Hendrycks and K. Gimpel, "Gaussian error linear units (gelus)," 2020, https://arxiv.org/abs/1606.08415.

[75] P. Ramachandran, B. Zoph, and Q. V. Le, "Searching for activation functions," 2017, https://arxiv.org/abs/1710.05941.

[76] D. Misra, "Mish: a self regularized non-monotonic neural activation function," 2019, https://arxiv.org/abs/1908.08681.

[77] R. Wightman, "Pytorch image models," 2019, https://github.com/rwightman/pytorch-image-models.

[78] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: unified, real-time object detection," 2016, https://arxiv.org/abs/1506.02640.

[79] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-cam: visual explanations from deep networks via gradient-based localization," in *Proceedings of*

the IEEE International Conference on Computer Vision, pp. 618–626, Venice, Italy, October 2017.

[80] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, “Learning deep features for discriminative localization,” in *Proceedings of the Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2921–2929, Las Vegas, NV, USA, June 2016.