An automatic inspection approach for remanufacturing components using object detection

by

Sri Ram Manidileep Aravapalli

A thesis submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Major: Industrial Engineering

Program of Study Committee:
Gül E. Kremer, Major Professor
John Jackman
Goce Trajcevski

The student author, whose presentation of the scholarship herein was approved by the program of study committee, is solely responsible for the content of this thesis. The Graduate College will ensure this thesis is globally accessible and will not permit alterations after a degree is conferred.

Iowa State University

Ames, Iowa

2022

TABLE OF CONTENTS

Page

LIST OF FIGURES

Page

LIST OF TABLES

Page

ACKNOWLEDGMENTS

I am heartily thankful to my committee chair, Dr. Kremer, for believing in me and giving me opportunities to be involved in her research work. Her constant encouragement, support at every stage of this research study is incomparable. I would like to thank my committee members, Dr. Jackman, and Dr. Trajcevski, for their guidance and support throughout the course of this research.

In addition, I would also like to thank Dr. Paul Kremer, Dr. Günay, and my research team colleagues who have constantly supported me throughout my research work with their valuable feedback and support. I am also grateful to my parents, Naveena Gorre, Sumanth Kaliki, Ashish Gorthy and Aravind Kalakuntla for all the help, support they gave to me. I would like to thank my friends, colleagues, the department faculty, and staff for making my time at Iowa State University a wonderful experience.

ABSTRACT

Remanufacturing is the process of restoring a used product to the specifications of original manufactured product with a matching warranty. This process benefits the remanufacturers to a greater extent as it just requires the replacement of worn-out or obsolete components, thereby providing significant economic, social, and environmental benefits. Remanufactured components should meet the customer's demand as new products and achieving this is a tough task due to uncertainty involved in the quantity and quality condition of the returned product. Inspection is one of the critical tasks in remanufacturing as it determines the quality of End of Life (EoL) product on arrival for remanufacturing, also inspection plays a crucial role in making the most appropriate decisions to proceed or scrap the product. Due to the drawbacks in the traditional manual inspection process because of its unreliable, subjective, time-consuming, and error-prone nature, attempts to adopt automatic inspection techniques gained attention in the industry. This study examines the capability of optical inspection techniques to increase productivity, profitability with higher reliability in remanufacturing. In the study, object detection methods are implemented to classify and locate defects on metallic surfaces using an open-source dataset GC10-DET. The detailed image pre-processing techniques, class imbalance techniques, and their effects on the model performance are also discussed. The YOLO (You Only Look Once) V4 algorithm with CSPDarknet-53 was used to locate and classify the defects. The performance of the algorithm is compared with other state-of-the-art techniques published in the literature for recall, average precision, and mean average precision (mAP) metrics. Our model demonstrates effective defect localization with a mAP of 66.78% and 93.63% for original and augmented datasets, respectively, which shows promise for the development of automated inspection technology.

Keywords

Object detection, defect detection, deep learning, remanufacturing, data augmentation, image pre-processing

CHAPTER 1. INTRODUCTION

1.1 Motivation

Inspection is the process of inspecting the parts to identify the defects and ensure that a produced part has all the features (geometric dimensions and tolerances) within the design specifications. It is usual practice that manufacturing units do the inspection process even though it is economically expensive and time-consuming to ensure that products meet customer satisfaction and safeguard the business reputation.

Montgomery et al. (1991) mentioned that inspection-free manufacturing is possible if the process capability ratio (PCR) is >= 1.33 for an existing process and >= 1.50 for a new process. Process capability ratio is the statistical measure of process capability, and process capability is the ability to produce the parts in specified tolerances. Mathematically PCR is represented as ((USL – LSL)/6sigma). USL is upper specification limit, and LSL is lower specification limit. Only a few companies achieved this inspection-free manufacturing (e.g., Hewlett Packard LaserJet printers) (Mital et al., 1998). However, most companies still include the inspection process due to instability or large variability due to the manual inspection.

There are three types of inspection processes a) Manual inspection, b) Hybrid inspection, c) Automated inspection (Kopardekar et al., 1993). As the name indicates, manual inspection purely depends on the human inspectors. Hybrid inspection is a semi-automated inspection process where some aspects of the inspection process can be automated and have the economic advantage compared with the fully automated process. Mital et al. (1998) stated that the hybrid method took less time with fewer errors (almost 47% fewer errors), and standard deviation of the process is better with hybrid method. The manual inspection process is labor-intensive, time-consuming, and has the scope to make the inspection process a bottleneck one (Intel, 2018). Intel

(2018) stated that it takes 6-9 months to train people for the manual inspection process to achieve the accuracy of 90 percent. Due to other factors like highly repetitive work, lack of insight, and process advancements, this accuracy can further decrease to 70-85%. Another drawback is that the manual inspection process results are not consistent and vary from person to person. To overcome these disadvantages, an attempt to implement a completely automated process using machine vision and machine learning techniques has been made in this thesis.

Nowadays, 'Sustainable growth' has become one of the important objectives to achieve economic growth without reducing the natural resources. Recycling, recovery, and remanufacturing are the three primary ways to support sustainable growth. Recycling is the process of converting waste products into new ones, and it reduces the need for raw material. Recovery is the process of removing good components from the disposed parts. Remanufacturing is the process of recovering the usable parts and recycling the unusable parts and reassembling the recovered parts into good products. There are five important units in remanufacturing: Product Acquisition, Reverse Logistics, Inspection and Disposition, Reconditioning, Distribution, and sales. Errington et al. (2013) stated that the European Union had passed legislation for certain product manufacturers (e.g., plastic), which imposes responsibilities to the makers to dispose of the products at the end of life. This shows us the importance of remanufacturing in today's industrial conditions.

Extensive research is being done on different aspects of remanufacturing like production planning approaches, operations management, reverse logistic channels, but little research is involved in the inspection process (Guide et al., 2002). Steinhilper et al. (1998) stated that assessing the condition of disassembled and cleaned parts is the most crucial step in

remanufacturing. It is a labor-intensive and time-consuming process to test, sort, and grade the returned products (Guide et al., 2002).

Inspection is one of the crucial aspects of remanufacturing to determine the returned item's condition. The process of accepting or rejecting the returned item is determined in this step. Sorting the components in remanufacturing 100% inspection is necessary to gain the second user's confidence and increase the manufacturer's profitability (Errington et al., 2013). Because of this 100% inspection, remanufactured products have better reliability than the new ones. However, when carried out manually, this inspection process can be very tedious. Hammond et al. (1998) stated that the lack of skill in inspecting the parts and the difficulty in identifying the defects as some of the major issues that inspectors experience. This thesis discusses the potential implementation of complete automation for the inspection process using machine vision and machine learning techniques with a case study.

1.2 Overview of Proposed Framework

The purpose of this study is to automate the inspection process using machine vision and machine learning techniques with great accuracy. Inspection process carried out in this study is to detect and classify the defects of metal components; and types of defects are discussed later in the section. Machine vision involves the utilization of vision systems to collect the images of the defects. Machine learning algorithms help to detect the defects of a component automatically without manual intervention. The aim of this study is to locate and classify the defects with great accuracy. The number of defects per image could be more than one. It involves the extensive review of existing literature for the methods used, accuracy achieved, and metrics used to

determine the performance of the algorithms used. The research objectives that frame the scope of this research are:

- Study the existing literature to collect the information to overcome the limitations involved in collecting the images and types of metrics to avoid bias and other factors of the algorithm.
- Collect the defect images for the case study to investigate and validate the proposed method. For the case study, dataset comes from the open-source data repository ‘Kaggle’/ ‘GitHub’.
- Identify and select the most appropriate image transformation techniques to augment the dataset.
- Implement the object detection algorithm on the dataset from case study to classify and detect the defect location.
- Tune the algorithm to choose the optimal set of hyperparameters for better model performance. Parameters that define the model architecture is known as hyperparameters and the process of searching the ideal model architecture is hyperparameter tuning.

1.3 Thesis Roadmap

The concept of automating the inspection process is not new; however, most discussed approaches are limited to classifying the defects where there is a single defect per image, or the performance of the algorithm is not sufficient to replace the manual inspection process. This study attempts to classify the defects which may not be visible to the human eye. Another

highlight of this study, apart from classifying the defects, is that to identify the exact location of defect.

Chapter 2 covers the literature review about remanufacturing, automated defect detection techniques used and the research gap that needs to be fulfilled. Chapter 3 explains the methodology, data pre-processing and information on the object detection algorithms considered in this thesis. Chapter 4 consists of the results. Chapter 5 features conclusions and limitations of the study. Further research scope is also provided.

CHAPTER 2. LITERATURE REVIEW

2.1 Introduction to Remanufacturing

With rapid changes in features and capabilities, the useful life of products becomes short, which causes an increase in the generation of waste. Globalization also contributes to depleting natural resources due to its wide reach because this reach creates the demand for mass production due to a more extensive consumer base. The depletion of natural resources causes adverse effects on humankind, and to combat this, the concept of sustainability is introduced. The three pillars of sustainability are economic viability, environmental protection, and social equity. With the growing popularity and increase of awareness about sustainability, manufacturers started to implement their part to achieve the goals of sustainability. Reuse, repair, refurbish, recycle, and remanufacturing are significant ways to support sustainable growth. If it is feasible, remanufacturing could be the best approach among all (Kandukuri et al., 2019). This is because of the lower price of the remanufactured part. Environmentally, remanufacturing promotes scrap reduction by producing the new products with the old items. Steinhilper et al. (1998) stated that remanufacturing could save up to 90% of material compared with the new part. Ijomah et al. (2007) stated that remanufacturing is preferable to recycling to achieve better profits. These economic and environmental benefits of the remanufacturing help to achieve the aim of sustainability.

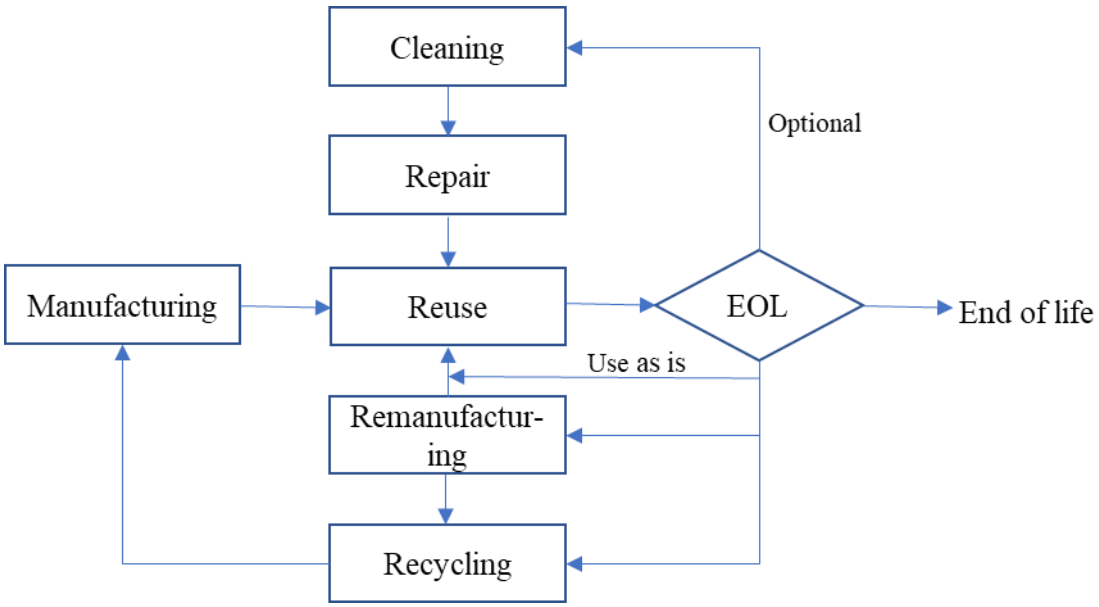

Figure 2-1 Remanufacturing as an alternative within the product life cycle (Kandukuri et al., 2019)

The Fig 2-1 illustrates the importance of remanufacturing, recycling, repair in the product life cycle. Despite having the advantage of reducing the lead times compared to ordering new parts, remanufacturing is very often misunderstood with the repair process. Fig 2-1 shows that the method repair directs the parts to reuse by increasing the life cycle little more. But remanufacturing recovers the functional and material value from the used product and remanufactured product performs the same way or even better than the original new products.

Remanufacturing refers to restoring a product to like-new condition by reusing, reconditioning, and replacing parts. Remanufacturing is also often confused with refurbishing and recycling. Refurbishing process brings the used product to a functional condition. The performance level may not be equal to the original product, whereas in remanufacturing, the performance level should be greater or equal to the new part. Recycling is just recovering the value of the material (Ortegon et al., 2013).

The reverse logistics of remanufacturing consists of the following stages (Ortegon et al., 2013):

1. Collection of the used product
2. Sorting
3. Testing
4. Disassembly
5. Cleaning
6. Reprocessing and part replacement
7. Reassembly
8. Inspection
9. Packing and transportation
10. Re-commercialization.

Each stage has its challenges. The process of collecting the used products, part yield, and remanufacturing efficiency are significant factors in deciding a remanufactured product's price (Sutherland et al., 2010). Traditionally, inspection is done after the complete disassembly and cleaning process. Only after inspection, the remanufacturing effort, time, and cost are determined. The cost of remanufacturing is volatile because the price of the remanufactured part depends on the quality of the returned product, product type, cost of material replaced. Also, the cost varies depending on the type of industry. For the electronics company, the disassembly and assembly process drives the overall cost of remanufacturing, and for the automotive industry, it is due to replacing a new part in place of a damaged or worn-out part.

Total cost of the remanufacturing can be written as (Ortegon et al., 2013):

$$RC = A + R_p + R_n + H + D \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (1)$$

Where RC represents total remanufacturing cost,

A is returned product acquisition cost,

R_p is cost of replaced components,

R_n is the reconditioning cost,

H is the Inventory cost, and

D is cost of disposal.

2.2 Defect detection techniques

Apart from manual inspection, defect detection approaches are divided into traditional recognition methods and deep learning methods. Traditional recognition methods consist of two steps: traditional image processing and machine learning techniques (Baumgartl et al., 2020). Traditional image processing techniques try to extract relevant data features from the raw images and these features are used to train the machine learning algorithm to learn the pattern in the images with target objects. Machine Learning (ML) techniques can be classified into three categories: supervised, semi-supervised and unsupervised (Baumgartl et al., 2020). In supervised learning, the data must be labelled as per their categories and fed to train the ML algorithm. For supervised learning, user must know, identify, and label the defects. In unsupervised algorithm data is used without labelling and the algorithm tries to predict defects by itself. Semi-supervised algorithm uses both labelled and unlabeled data and involves the mixed approach of supervised and unsupervised training method.

Traditional image processing techniques based on the image background texture and divided into four categories: structural-based, statistical-based, filter-based, and model-based

methods (Ren et al., 2017). Structural-based methods are used in the applications like textile, fabrics where repetitive patterns exist. Important structural methods widely used in the industry are edge features, skeleton representation, morphological operations, and primitive measurements. Statistical methods are efficient for stochastic textures like ceramic tiles, wood, and castings. This method is based on the distribution of pixel values in the image and some of the popular statistical methods are the histogram-based method, local binary pattern, and co-occurrence matrix. Filter methods are the ones that apply filters to the image texture. The type of filter method used depends on the domain of the application, and some of the filter methods are spatial-domain, frequency domain, and spatial frequency domain. Model-based methods uses multiple properties of the defects for constructing the representations of the image. Gaussian mixture entropy model comes under model-based methods.

Ren et al. (2017) stated that these traditional image processing techniques aim to construct features for images, but there were no clear guidelines for selecting the optimal method. Since the metallic surfaces are prone to illumination and background clutter, these traditional image processing techniques cannot be implemented directly to the metallic surface. Also, in applications of these methods, depending on environmental factors, some parameters need to be adjusted, and even there are times where the whole algorithm needs to be re-designed completely. These approaches generally aim at only one specific environment, which is difficult to deploy in the more challenging real-world due to the lack of robustness and adaptability. Another important factor is that machine learning algorithm will not learn any pattern if the image processing techniques miss any important features and the chances of missing the important features is very high using the manual, traditional methods. With these disadvantages of getting suboptimal results led to the introduction of deep learning techniques for extracting the

image features instead of constructing the images. The deep learning networks have the advantages of automatically generating the relevant features and classifying the objects without intervention of any manual image processing technique. CNNs convert the raw image data to meaningful data representation - ‘feature maps’ which supports the final classification (Baumgartl et al., 2020).

Deep learning techniques have two challenges in their implementation. One is the dataset, and the other is the precise identification of the defect area. Collecting a large amount of dataset is costly and need tedious manual work to label the data. Ren et al. (2017) proposed a generic method that classifies the defects using a small dataset, and patches were extracted from the pre-trained network, and predictions were made using the heat map.

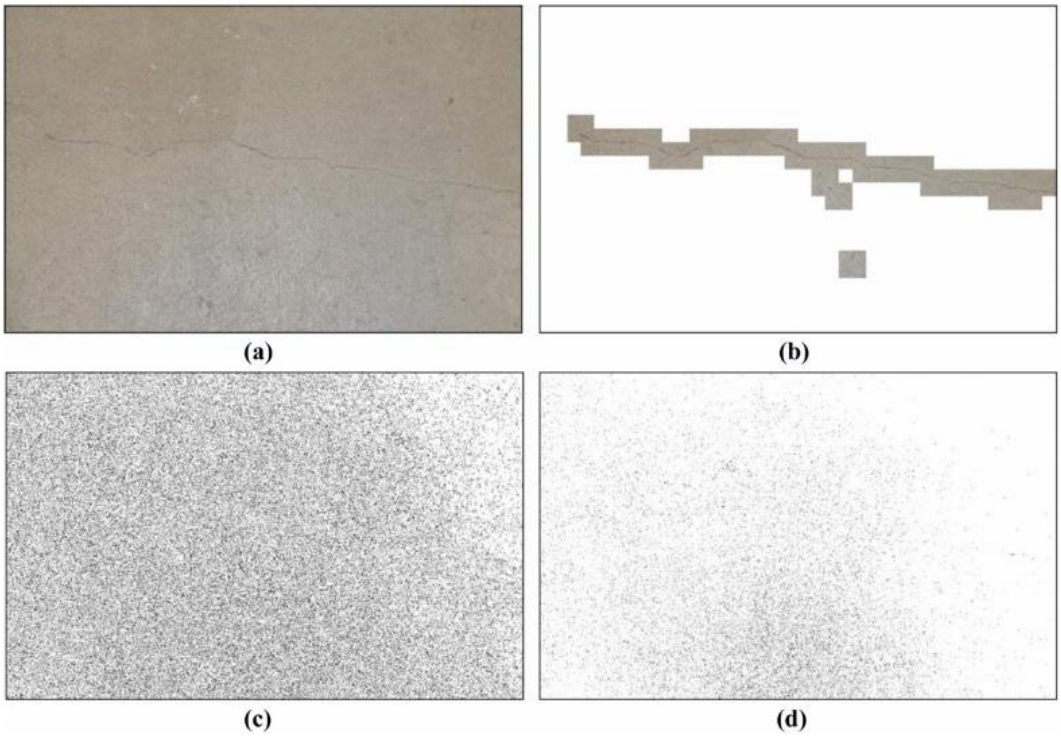

Figure 2-2 Crack detection using sliding window technique
(a) Original image, (b) Proposed CNN, (c) Canny edge detection, and (d) Sobel edge detection (Cha et al., 2017)

Cha et al. (2017) proposed implementing a convolutional neural network and sliding window technology to detect the concrete cracks. The proposed network is trained on 40,000 images with 256 x256 resolution and achieved validation accuracy of approximately 98%. Reflective and shadow defects were classified with great accuracy. From the figure 2-2, we can see that the proposed CNN method gives the accurate results of the thin crack defects but the traditional methods like Canny and Sobel edge detection both failed to detect the crack defect. The performance of the traditional methods was dependent on various condition like lighting, brightness, contrast of the image. But the proposed method using CNN produced efficient results irrespective of any conditions of the image (Cha et al., 2017). This method uses the sliding window technology, which is computationally expensive and not feasible in real-time tasks. The main drawback of this method is that it needs large amount of training dataset that is practically expensive and impossible to get in many cases.

Lv et al. (2020) proposed implementing an end-to-end detection network (EDDN) using a single shot multi-box detector. The author implemented the algorithm on two datasets: the public NEU dataset and his own GC10-DET metallic surface defect dataset. Comparisons with the existing methods like SSD, Faster RCNN and YOLO V3 were made, and it was stated that the proposed method achieved better accuracy with less computation cost.

Rahaman et al. (2009) proposed a defect detection and classification technique using ceramic tiles. The author used the traditional image processing techniques and achieved the classification accuracy of almost 90% in all the classes, but the proposed method was tedious and time-consuming, which is not feasible in real-time situations. The method also failed to detect the glaze and the scratch defects.

Yun et al. (2020) proposed convolutional variational autoencoder (CVAE) to generate enough data for the deep learning network and used these images for training the deep convolution neural network to classify the metallic defects. Testing was performed using the images collected from actual metal production line and an accuracy of 99.69 % was achieved, but while using this CVAE technique performance of the algorithm varies with respect to number of images used for CVAE to generate extra samples. Also, the performance dropped by almost 6% when a smaller number of samples were used for CVAE. There is a high chance of getting more false positives if there is any ambiguity in the dataset.

Essid et al. (2018) proposed autoencoder deep neural network architecture to classify and locate the manufacturing defects. The autoencoder trained on the images obtained from different feature extraction techniques and comparison was made using the classification method like KNN, SVM. Though there was an improvement in terms of false positives and false negatives, it was very minimal and may not be reliable in the real time situations. Also, this method needs additional feature extraction process, which may be computationally not feasible.

Lin et al. (2021) studied the impact of quality of training images on the performance of the object detection algorithms and found that defect visibility and over exposure had a significant impact on the algorithm performance and this brings the importance of image pre-processing techniques to match all possible conditions in the training dataset that can occur in the shop floor.

Implementation of pretrained network using metallic defects is one of the complicated tasks due to the non-availability of thousands of data points, and defects may not be visible due to surface quality and reflective property of the metallic surface. The current research shows a gap in the detection and localization of metallic defects on uncleaned surfaces. Recent studies

used traditional methods, sliding windows, or single-shot multi-box detectors. There is a need for developing the most efficient algorithm which performs better in terms of accuracy while being economically viable. An advanced technique, YOLO v4, is used in this thesis on an open-source dataset. The dataset is publicly available (GC10-DET); it was collected in real time production shop floor (Lv et al., 2020). The comparative study is conducted using different industry techniques, and accuracy with false negatives and false positives are reported.

2.3 Convolution neural network

Convolution neural network is a feed-forward neural network and well known for its applications in speech and image-data analysis. CNN uses the convolution and pooling layers to transform the image into its essential features and with the help of these features, network classifies the image. CNN usually consist of four major layers (see Fig 2-3): 1) convolutional layer; 2) activation layer; 3) pooling layer; and 4) fully connected layer. Convolution layer uses filter, also known as kernel, to view the few pixels of the input at a time and obtains the feature maps of the input image. In convolution process, kernel goes over the input image and performs the matrix multiplication. Convolution operation is a dot product of the original pixel value with weights defined in the filter. Let us consider W_i the weight of filter i, b_i is the bias of filter i, X_s be the small part of the image with pixel intensity values of the image, σ is the activation function, and the size of input image is $M \times N$. The convolution of X_s with filter i is obtained as (Ren et al., 2018):

$$f_{i,s} = \sigma(W_i X_s + b_i)$$

This process goes over the whole image using sliding mechanism with the patch size of a×b and stride size s, the size of the final output after convolution process is obtained as

$$k \times \lceil [(M - a + 1)/s] \rceil \times \lceil [(N - b + 1)/s] \rceil ,$$

where $\lceil . \rceil$ is the ceiling function & k is the number of filters (Ren et al., 2017).

The activation layers help to induce non-linearity to allow the network to train through back propagation. ReLu, Tanh, Sigmoid are some of the activations that are widely used in the industry.

The pooling layer is the process of down sampling and reduces the size of the matrix. In this process, only one value is taken from the group and what is shown below represents max. pooling, where maximum values in the group is being picked.

$$Pool_s = \max (X_s)$$

The pooling operation helps to train the network much faster by focusing only on the critical and important features of the image. This helps to avoid the over fitting issues in the network. For example, the output of the max pooling process with input size $M \times N$ and patch size $c \times d$ is obtained with a size $\lceil [(M - 1)/c] \rceil \times \lceil [(N - 1)/d] \rceil$ where $\lceil . \rceil$ is the ceiling function.

Fully connected layer has the structure like the traditional multilayer perceptron. Fully connected layer is obtained after flattening the output of its previous pooling or convolutional layer and represents in the form of a one-dimensional vector. Flattening is the process of unrolling all the values in a matrix into a vector. These fully connected layers uses the softmax activation function to get the probabilities. Output of the fully connected layer contains a list of probabilities of the input being in a particular class and classification decision is made based on these probabilities.

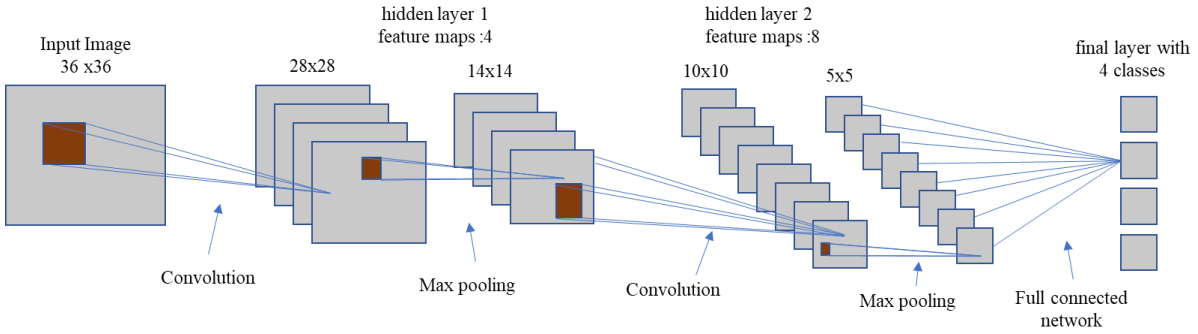

Figure 2-3 Detailed picture of the convolution neural network

2.4 Deep neural networks

In deep learning, CNN is the most representative model, and the typical architecture is generally referred to as VGG16. The architecture of VGG16 is shown in the figure below. Each layer in the CNN architecture is a feature map and the input layer feature map is a 3D matrix of pixel intensities of different color channels. Each neuron is connected to a portion of neurons in the previous layer, and this is known as receptive field. The feature maps can undergo different types of transformations like filtering, pooling. The final layers in the CNN architecture will be several fully connected layers and these layers will have different activation functions to calculate the conditional probability for each output neuron. From the above-mentioned components VGG16 has a total of 13 convolutional layers, 3 fully connected layers, 3 max-pooling layers and a SoftMax activation function to the final layer to calculate the conditional probabilities.

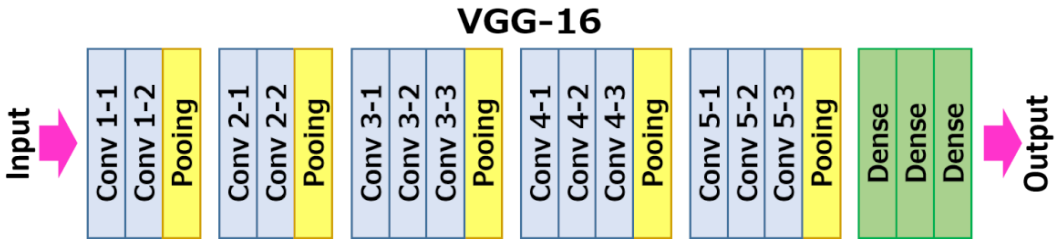

Figure 2-4 Architecture of VGG16 network.

Starting with Alexnet, deep neural networks showed their superiority over other techniques in the ImageNet data competitions, and this showed the importance of network depth and the depth of the networks increased from sixteen to thirty in consecutive years in the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) competition. In view of the results from ILSVRC competition, a debate on the significance of depth for learning within networks has started. He et al. (2016) stated that with the increasing number of layers, the algorithm does not always give the better results; and additional layers can also deteriorate the performance due to vanishing/exploding gradients and the degradation problem. The phenomenon of the accuracy reaching the saturation level and then rapidly degrading with the increase in depth is known as the degradation problem.

To address this issue of vanishing gradient and degradation problem, He et al. (2016) proposed a residual learning framework with skip connections. Fig 2-5 & 2-6 shows the training error and validation error using the plain and ResNet networks. In the case of plain network, it is evident that training and validation error is higher using the network with 34-layers than the one with 18-layers. But with the ResNet, the deep layered network (34-layer) performed better than the shallow network (18-layer). Also, error percentage is very less comparatively using the ResNet than the plain network. Fig 2-7 shows the feed forward network with shortcut connections (skip connections) at each layer. In ResNet, the output of the skip connections gets added to the outputs of the stacked layer. Mathematically, the output of ResNet at each layer can be written as $H(x) = F(x) + x$ where $H(x)$ is the desired underlying map, $F(x)$ is the output of the stacked layer and x is the identity mapping.

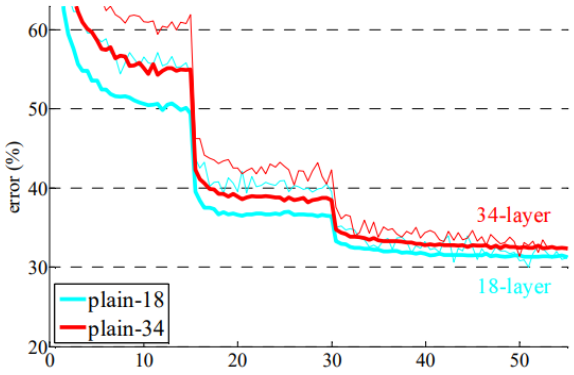

Figure 2-5 Thin curve indicates the training error and bold curve indicates the validation error using the plain network (Adopted from He et al., 2016)

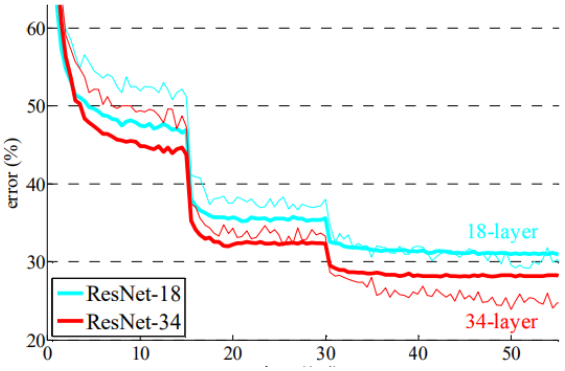

Figure 2-6 Thin curve indicates the training error and bold curve indicates the validation error using ResNet (Adopted from He et al., 2016)

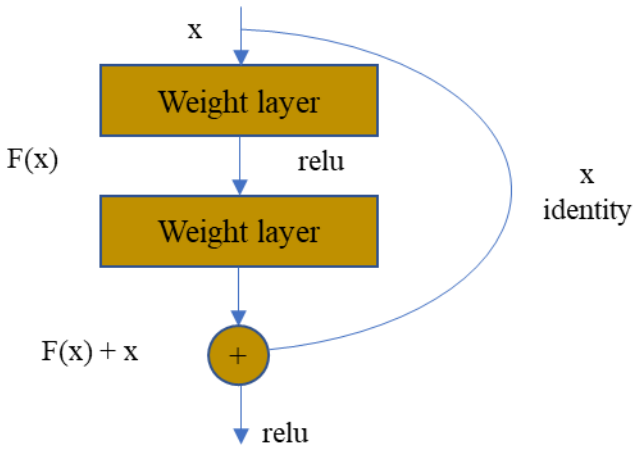

Figure 2-7 Residual learning framework with shortcut connections (Adapted from He et al., 2016)

He et al. (2016) stated that with the help of ensemble of these residual network frameworks, an error rate of 3.57% was observed on the ImageNet test set and won the 1st place in the ILSVRC 2015 classification task. Ensemble is the process of using multiple learning algorithms together to get the better results by combining output from all the algorithms.

2.5 Feature learning from pretrained network

To train a very deep and complex CNN from scratch, one needs a hundred of GBs of RAM and GPUs. Although little investment is required to get the RAM, GPUs are expensive, and to select an optimized GPU with better cost and performance requires lots of effort. Also, a large amount of data to train the network from scratch is needed. To overcome this issue, many researchers use the pretrained network whose network was already trained with a large-scale image dataset like ImageNet and uses the same pretrained weights as a feature extractor for the other data. This pre-trained network needs just a small modification to replace the last few layers of the pre-trained CNN as per the number of classes.

2.6 Size of the training data

Articulating the problem early is the important step to decide what type of data need to be collected and which data will be more valuable. Acquiring the training data for machine learning or deep learning models can be an expensive task as it involves significant manpower, equipment run time and licensing fees etc. Therefore, it is always a critical step to obtain huge amount of training data to achieve a specific model performance. The training data size issue is also known as sample complexity. For image classification using deep learning, a rule of thumb for the number of images required for training is 1,000 images per class and this number can go down

depending on different criteria such as using the transfer learning technique with the same domain and images with great quality (Warden, 2017).

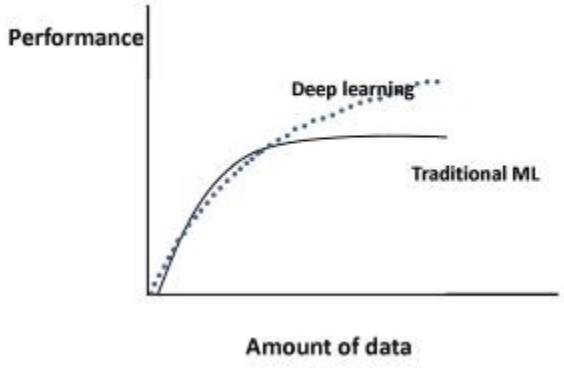

Figure 2-8 Performance of the algorithm with the quantity of training data (Adopted from Mista 2019)

The above figure (Fig 2-8) shows the change in performance of machine learning algorithms with increasing data size in the case of traditional machine learning and deep learning algorithms. For both traditional machine learning (traditional ML) and deep learning algorithms, the performance of the algorithms grows according to a power law but in the traditional ML case the performance reaches a plateau. Sun et al. (2017) stated that performance of the deep learning techniques increased logarithmically with increasing training data size. On the other hand, Joulin et al. (2016) stated that performance of the convolutional networks using a dataset of 100 million Flickr images and captions flattens after 50 million training images. Lei et al. (2018) stated that image classification accuracy increases with training data size, but the model robustness started to decline after a certain model-dependent point.

For determining the optimum size of the training dataset to achieve high classification accuracy with low variance, Cho et al. (2015) used the learning curve method which can be

represented as an inverse power law function and has the ability to check the variance, bias, underfitting, overfitting using the single graph between MSE and training size.

Bias and variance are the two important metrics for measuring the error of the model. A low bias model fits the training data very well and change in training error will be significantly high with change in dataset and this change in error represents the variance involved in the model. Low bias model will have high variance and vice versa. These two metrics play an important role in deciding the size of the dataset. Fig 2-9 illustrates that output of the model ($\hat{f}$) changes with change in training data which indicates that model is having low bias and high variance. Fig 2-10 illustrates that output of the model ($\hat{f}$) is constant with change in training data, but the output failed to fit all the points in the model, and this indicates that model is having high bias and low variance.

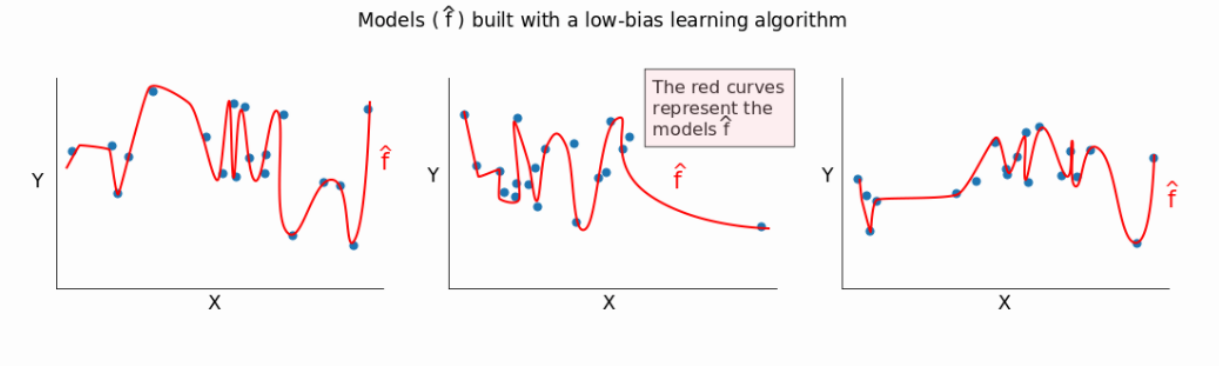

Figure 2-9 Different training data gives different output (redline) which shows the low bias and high variance in the model (Adopted from Olteanu 2018)

Figure 2-10 Different training data gives similar output (redline) which shows high bias and low variance in the model (Adopted from Olteanu 2018)

From the above two graphs, one can say that it is not possible to have low bias and low variance at once. One should choose the model with optimum values of both bias and variance.

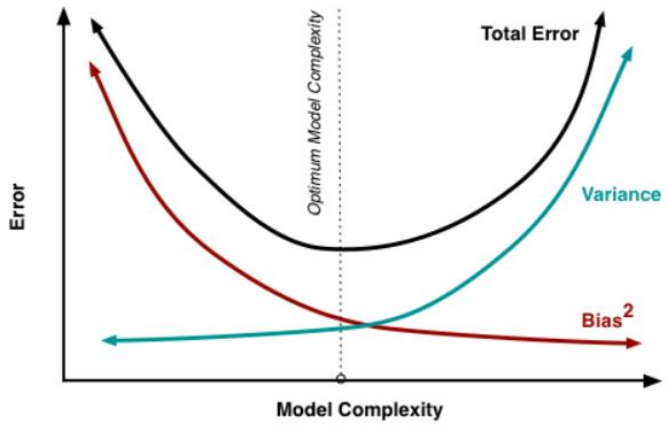

Figure 2-11 Model complexity with optimum values of bias and variance (Adopted from Olteanu 2018)

Fig 2-11 illustrates the relationship between bias and variance with respect to the model complexity and error. If the model is simple, then bias in the model will be high with low variance and vice versa. Also, training the model with huge amount of data makes the model becomes less bias and thereby increases the complexity of the model. For the Machine learning models, bias and variance should be at optimum level to have less error, better performance and

less complexity. The optimum size of the training data can be estimated using the learning curve as described below. In any dataset, the minimum size of the training data is one. But using a single sample, the validation error (green line in the below graph) will be high. The maximum size of the training dataset can be the whole dataset. From the fig 2-12a, it is evident that using the whole dataset is not a better option as both the training and validation curves plateau at some point and therefore, it increases the computational time unnecessarily. In fig 2-12b, we can see that there is good scope to further converge the validation curve towards the training curve. In this scenario adding more data helps to get the best results. The optimum size of the training dataset will be the point where the gap between training and validation curve is minimum i.e., low variance. The gap between two curves indicates the variance. Adding more data beyond that point will not help the model to further increase the accuracy. Another important observation in the graph is the error value. If the error value is higher, it indicates that training error is larger with higher bias. The training size should be selected according with optimum bias and variance values as shown in the figure below. Depending on the difficulty in collecting the dataset, data augmentation techniques like photometric, geometric, or generative adversarial networks (GANs) will be implemented to increase the dataset.

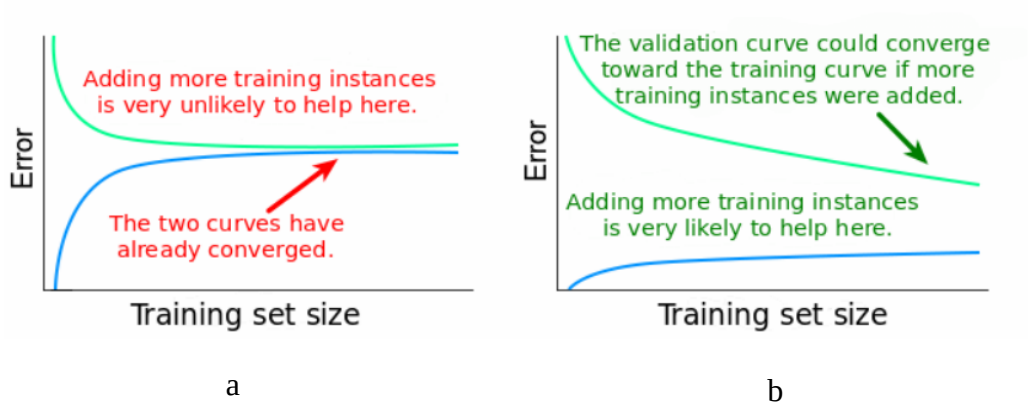

Figure 2-12 a) High bias and b) low bias scenarios that guides to select the size of training dataset (Adopted from Olteanu, 2018).

2.6 Object detection

Object detection is the technique which not only performs the task of classifying different images but also detects the position of the objects (object localization) contained in each image. It also helps to provide important information related to images and videos which solves many computer-vision tasks. Object detection has not evolved in a single day. It started with the help of handcrafted feature extraction techniques and shallow networks. In general, pipeline of an object detector is classified into three categories: Informative region selection, feature extraction and classification (Zhao et al., 2019).

Informative region selection is the process of scanning the whole image with a multi-scale sliding window to find all the possible positions of the objects. Zhao et al. (2019) stated that the process of scanning the images is computationally expensive and produces too many unwanted results. This process can also lead to unsatisfactory results in case any constraints imposed on the sliding window templates.

Feature extraction is the process of extracting the visual features that give the semantic and robust representations. Scale invariant feature transform (SIFT), Histogram of oriented gradients (HOG) and Haar – like features are some of the traditional feature extraction methods. They have drawbacks due to their high dependency on the illumination and background conditions.

Classification is the task of distinguishing the objects into different categories and this can be achieved with the help of classifiers like Support Vector Machine, AdaBoost, Deformable Part-based model and these classifiers provide better hierarchical, semantic, and informative representations.

Though with the help of these local feature descriptors and shallow networks have less burden on the hardware, only small improvements have been obtained with the minor changes of

successful methods until the emergence of deep neural networks (DNN) (Girshick et al., 2014). The reasons for the small improvements and their inferiority to DNN is due to the redundant sliding window strategy, which is inefficient and inaccurate, and difficulties with the feature extraction process due to their high dependency on manual operations.

Zhao et al. (2019) stated that a greater improvement in performance was achieved with the introduction of Regions with CNN features (R-CNN). Deep neural networks differ from the traditional and shallow networks in such a way that they have deeper architecture that helps to learn complex features than the traditional ones. Significant gain in performance of the model was observed since the proposal of R-CNN. Faster R-CNN optimizes both the classification and bounding box regression tasks, and YOLO performs the object detection task with the help of fixed-grid regression.

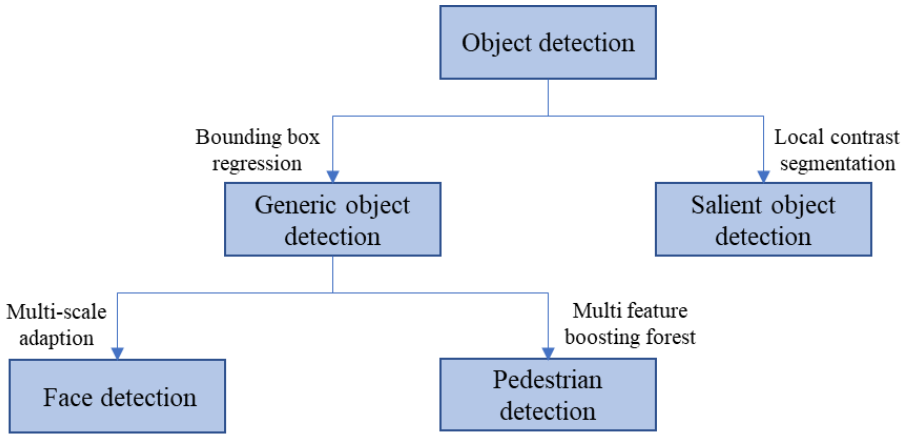

Figure 2-13 Applications of object detection (Adapted from Zhao et al., 2019)

Fig 2-13 illustrates that the task of object detection is categorized into two parts: 1. Generic object detection, and 2. Salient object detection as shown in the above figure. Generic object detection is achieved with the help of bounding box regression whereas the latter one

depends on the local contrast enhancement and pixel-level segmentation tasks. The generic object detection model was used in this study to detect the defects on the metallic defects.

2.7 Generic object detection

The goal of the object detection model is to locate and classify the existing objects in the image with the help of bounding boxes and their confidence score. Generally, the generic object detection model was classified into two categories based on their approaches to detect the objects. One category first generates the region proposals and then classifies the proposal into different categories. The second one performs both region proposals and classification task in a single step to achieve the results. The first category is termed as a two-stage object detector and the second is known as a one-stage object detector. The two-stage object detector includes methods like R-CNN, SPP-net, Fast R-CNN, Faster R-CNN, R-FCN, and mask R-CNN. The examples for one stage object detectors are YOLO, SSD, DSSD, DSOD, Multibox.

2.7.1 Two stage object detectors

In this process, coarse scan of the whole image is carried out first and then the algorithm focuses on the region of interest.

2.7.1.1 R-CNN

Girshick et al. (2014) proposed the R-CNN method to improve the quality of bounding boxes with the help of deep architecture to extract the high-level features. R-CNN process is divided into three stages: 1) Region proposal generation, 2) CNN based deep feature extraction, and 3) Classification and localization. As shown in the fig 2-14, region proposal algorithms like selective search method is used to generate proposal that contains the target object, and these

were fed to the convolutional neural network to extract the features from the proposals. With the help of linear support vector machine algorithms (SVMs), the extracted features were classified into their respective classes with a confidence score which were later produced with bounding boxes using the bounding box regressor.

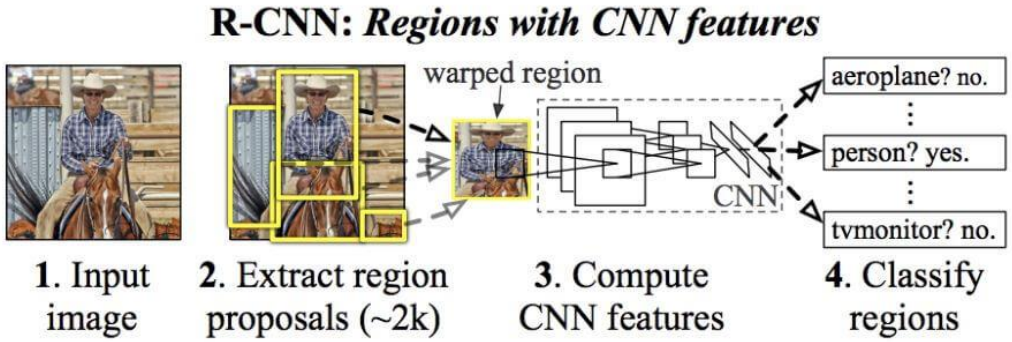

Figure 2-14 Architecture of R-CNN (Adopted from Girshick et al., (2014))

The region proposal method uses the image as an input and generates bounding boxes to all patches in an image which are most likely to be objects. These bounding boxes can be noisy, overlapping and may not exactly contain the objects. However, among the region proposals, there will be one that is very similar to the actual image's object. These proposals can be classified using an object classification model. Region proposals with high confidence scores correspond to object locations. Region proposal method should have very high recall rate because it is okay to have some false positives rather than missing true positives. Objectness, Constrained Parametric Min-Cuts, Randomized Prims, and selective search are some of the region proposal methods. The most popular method of selection is selective search because it is fast and possesses a high recall rate.

Selective search is based on the process of grouping similar pixels based on their color, texture, size, and shape using hierarchical clustering. This method starts by over-segmenting the image using graph-based segmentation based on pixel's intensity. The fig 2-15 shows the region proposals of the input image at different scales using the selective search algorithm. This region proposal method generates 2000 different regions for each individual image, which are still very small compared to the sliding window approach.

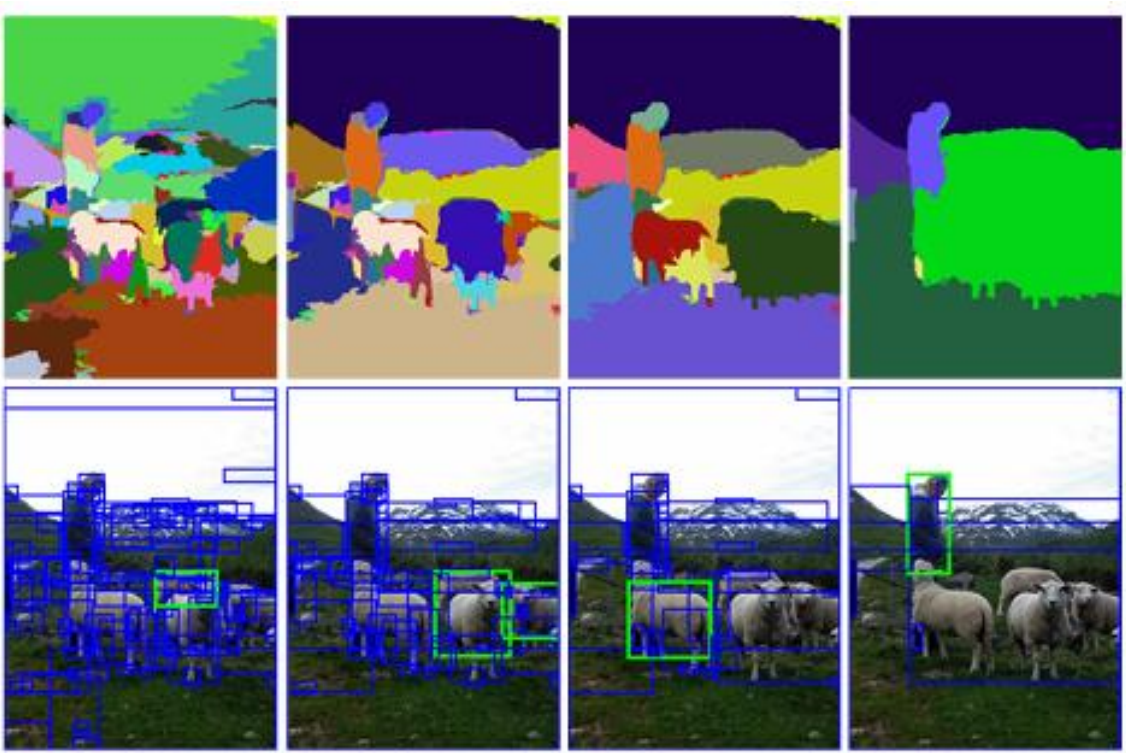

Figure 2-15 The region proposals of selective search algorithm at different scales (Uijlings et., (2013))

A large convolution neural network is used to extract the fixed length feature vectors from each region proposal and classifies each region using class specific linear support vector machines. CNN can be selected based on the application. Bounding box regressor is used to generate the bounding box to locate the target objects. Girshick et al. (2014) stated that it can

significantly reduce the dominant error mode like mis-localizations. Girshick et al. (2014) stated that R-CNN achieved the mAP value of 31.4% which is a large improvement compared to the Over Feat model which had the previous best results of 24.3%.

2.7.1.2 Faster R-CNN

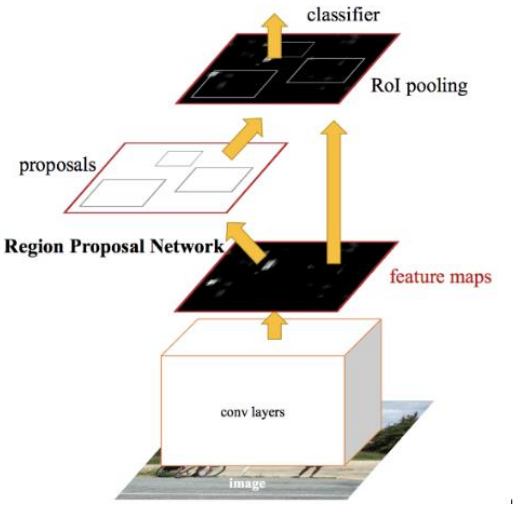

Figure 2-16 Unified architecture of Faster R-CNN (Adopted Ren et al., 2015)

According to Ren et al. (2015), convolutional feature maps used by region-based detectors, like R-CNN & Fast R-CNN, can also be used for Faster R-CNN to generate region proposals. The Faster R-CNN object detection network consists of a feature extraction network, which is typically a pretrained CNN, just as like its predecessor like Fast R-CNN. The Region proposal network (RPN) was constructed with the help of additional convolutional layer on the top of the feature extraction network (pretrained CNN), thus makes the RPN a type of fully convolutional network that can be trained end-to-end for generating proposals for detections.

Faster R-CNN consists of two parts: 1) Region proposal network (RPN), which generates object proposals and scores at each location simultaneously, and 2) Fast-RCNN detector for predicting the actual object class. Fig 2-16 shows the unified architecture of faster R-CNN. From

the fig, it is evident that image is used as an input for the convolutional layers which were common for both RPN and detector. RPN uses the input as feature maps from the convolutional layers and produces rectangular object proposals as output along with the confidence scores for each one of the proposals. After getting the region proposals from RPN, ROI pooling and upstream classifier and bounding box regressor will be used to produce the desired prediction results. With the help of this RPN technique, one can avoid the computationally expensive methods like Selective Search.

In short, RPNs share the convolutional layers along with the state-of-the-art object detection networks as shown in the fig 2-16. With the help of Faster R-CNN, region proposals can be generated in10ms per image. Another advantage of the RPN is that it generates the region proposals with a wide range of scales and aspect ratios using anchor boxes (Ren et al., (2015)).

2.7.2 One stage object detector

2.7.2.1 YOLO v4

YOLO v4 is one of the popular one stage object detectors that was developed to improve the accuracy and the speed of previous YOLO models. It is the latest and the most advanced version of YOLO algorithms and capable of producing 12% faster and 10 % more accurate detections than YOLO v3 (Bochkovskiy et al., 2020). YOLO-V4 boosts the prediction accuracy via a deeper and complex network architecture, where dense blocks (Huang et al., 2017) are used. In a dense block, each layer is connected to all other layers in the network. The feature maps of all the preceding layers along with its own feature maps are used as the inputs to all the subsequent layers. The architecture of a dense block is shown in below figure 2-17.

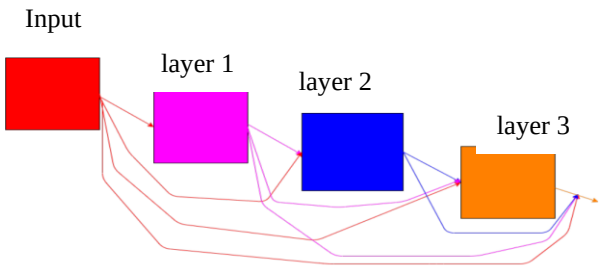

Figure 2-17 Dense block. All the preceding feature-maps taken as input for each layer.

The use of Bag of Freebies (Bochkovskiy et al., 2020) and Bag of Specials (Bochkovskiy et al., 2020) strategies also account for better performance. Bag of Freebies strategies improve the performance by changing the training process without increasing the inference cost. These strategies include applying data augmentation methods, solving the semantic distribution bias in the dataset, and using Complete Intersection over Union (CIoU) (Zheng et al., 2020) loss function in the objective function of the bounding box regression. Complete IoU is based on the three geometric factors like overlapping area, distance and aspect ratio and it is proposed by imposing the consistency of aspect ratio. A data augmentation technique increases the variability of the input images and helps to improve the performance of the model. Data augmentation can be done in terms of photometric distortions and geometric distortions. Photometric distortions deal with image aspects such as brightness, contrast, hue, saturation, and noise. Geometric distortions deal with random scaling, cropping, flipping, and rotating. Cutout (DeVries et al., 2017), Random Erase (Zhong et al., 2020), MixUp (Zhang et al., 2017), and Mosaic augmentation techniques (Bochkovskiy et al., 2020) can also be used for improving the robustness of the model. Semantic distribution in a dataset may have bias due to class imbalance in the dataset and this can be addressed using techniques like hard negative mining. A CIoU loss function enhances the convergence accuracy and speed of solving the Bounding Box regression problem as it adjusts the predicted bounding box closer to the ground truth box. The Bag of

Specials strategy significantly improves the accuracy of the model with an increase in the inference time (Bochkovskiy et al., 2020).

2.8 Overview of the literature review

From the table 2-1, we can see that there is at least one drawback in all the methods in terms of either computational or accuracy of the model. Tao et al. (2018) successfully segmented the metallic defects, but no information is provided about training and inference time. Also, the accuracy is less than 90%. Essid et al. (2018) proposed autoencoder deep neural network to classify and detect the metal defect. There is no information provided on inference time and metrics like average precision, recall mAP. Gai et al. (2020) proposed modified VGG network, but the performance is limited to 77% which does not hold good for real time practices. Cha et al. (2017) proposed detection method using sliding widow technique which is computationally expensive. Natarajan et al. (2017) proposed a deep neural network method on metal surface defects with better accuracy, but the method was limited to classification task only. There is a large gap for detecting and classifying the metal defects with good accuracy while keeping computations inexpensive. In this study, YOLOV4 is considered for better mAP values and its computationally feasible methods to implement in real time applications.

Table 2-1 Overview of literature review

Authors	Type of Inspection	Architecture used	Type of defects	Achievements	Limitations
Tao et al. (2018)	Automatic inspection	A novel cascaded autoencoder (CASAE) for localization & CNN architecture as classifier	Metallic surface defects	Successfully segmented the metallic defects under complex lighting conditions and ambiguous defects	Information about training and inference time not mentioned and accuracy is less than 90%
Essid et al. (2018)	Automatic inspection	Autoencoder deep neural network to learn features and Gaussian process as classifier	Manufacturing defects in metal boxes	Proposed network outperforms traditional techniques based on hand crafted features. False positives and false negatives were about 10% & 5% respectively.	There is no information provided on inference time and metrics like average precision, recall, mAP
Gai et al. (2020)	Automatic inspection	Modified VGG network (used ResNet and Inception ideology for one layer)	Metal surface defects	Obtained better results in comparison with VGG foundation model.	The mAP of 77% is still less for real time inspection.
Cha et al. (2017)	Automatic inspection	Deep architecture of CNN	Concrete cracks	Successfully detected cracks (98% accuracy) without any IPTs for extracting features.	Used sliding window technology which is computationally expensive
Baumgartl et al. (2020)	Monitoring the printing process	Depthwise-separable CNN	In process defects like splatter, delamination	Defects detected with an accuracy of 96% with low computational cost.	Investigations on defects like cracks, pores, bailing yet to evaluate
Yun et al. (2020)	Automatic inspection	CVAE and deep CNN based algorithm	Metal surface defects	Successfully implemented CVAE method to deal with class imbalance and insufficient data	CVAE crops the defects and can provide mislead information in case of similar defects

Table 2-1 Continued

Authors	Type of Inspection	Architecture used	Type of defects	Achievements	Limitations
Lv et al. (2020)	Automatic inspection	EDDN using single shot multi box detector	Metallic surface defects	Better performance in comparison with SSD, Faster R-CNN, YOLOv2 & YOLOv3	Achieved mAP does not suit for real time detection in the shopfloor
Yeum et al. (2015)	Automatic inspection	Object detection and grouping technique	Cracks on bridges	Detected cracks on difficult to access bridges with an accuracy of 98.7%	Further validation on defects like pinholes yet to make. computationally not effective.
Masic et al. (2013)	Semi-automated inspection	Multi scale pyramid pooling network	Steel defects	Classified steel defects with variable input sizes	Limited to only classifying the defects
Natarajan et al. (2017)	Semi-automated inspection	Flexible multi-layered deep feature extraction framework based on CNN	Metal surface defects	Outperformed traditional handcrafted features and achieved better accuracy	Limited to only classifying the defects and may not be helpful for large datasets.

CHAPTER 3. METHODOLOGY

3.1 YOLO v4

The overall architecture of YOLO-V4 used in this study presented in the figure 3-1.

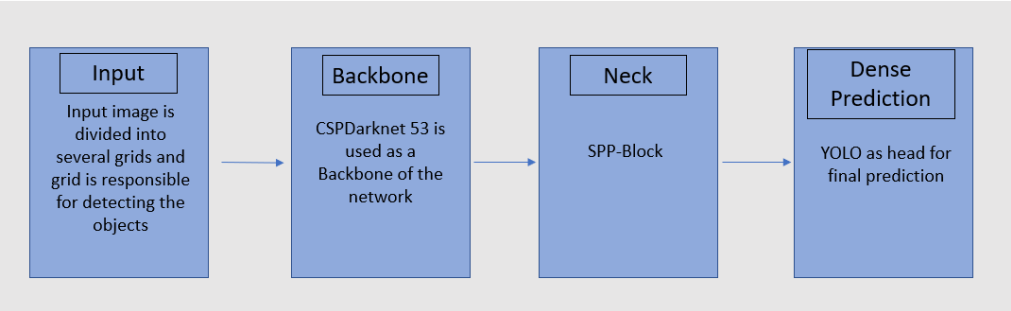

Figure 3-1 Architecture of YOLO-V4

The input image is divided into $S \times S$ number of grids and CSPDarknet53 (Cross Stage Partial Darknet53) is used as a backbone for feature extraction. CSPNet (Wang et al., 2020) helps to achieve a richer gradient combination with reduced computation time and increased speed and accuracy. The CSPDarknet53 network with its deeper layer can produce different levels of features with its higher semantics. Instead of concatenating the output with its input, CSPDarknet53 divides the input into two parts: one part goes through the network and other part concatenates with the output of the other half. The SPP-block (Spatial Pyramid Pooling) (He et al., 2015) serves as the neck part of the architecture and generates fixed size features irrespective of the size of the feature maps using the max-pooling technique. The SPP is selected because of its multi-level spatial bins and its capability to extract features at variable scales (Bochkovskiy et al., 2020). YOLO is used as a head for dense predictions. Dense prediction is a vector that contains the coordinates of the predicted bounding box, confidence score of the prediction, and the label.

The defect detection process for YOLO-V4 is shown in figure 3-2 for the GC10-DET defect dataset. The detection initiates with original input image shown in the figure 3-2a. Then the image is divided into $S \times S$ grid as seen in figure 3-2b. The grid cell is responsible for detecting the defect if the center of the defect falls into that grid cell. These detections were represented in the form of 'B' bounding boxes and the respective confidence score of each bounding box (see figure (c)) along with the object class. A confidence score indicates how confident the algorithm makes the prediction. The confidence score of a bounding box C, is calculated as:

$$\mathrm{C} = \Pr(Class_i|Defect) \times \Pr(Defect) \times IoU_{Pred}^{Truth}$$

where $\Pr(Class_i|Defect)$ represents the probability of the defect inside the box belongs to $Class_i$ type of defect. $\Pr(Defect)$ is the probability of having the center point of the defect in the grid cell, and IoU_{Pred}^{Truth} refers to the degree to which the bounding box interacts with the ground truth box.

(a) original image (b) S x S grid (c) Detection result

Figure 3-2 *Detection process of YOLO-V4*

3.2 Performance metrics

To evaluate the algorithm's performance, we used the mean average precision, recall, and average precision metrics. Generally, for object detection, the concept of intersection over union

(IoU) is used, and it is defined as the ratio of intersection of two bounding boxes (ground truth and prediction box) to their union. In this study, an IoU threshold value of 0.5 was considered. For bounding boxes with an IoU > 0.5, the bounding box is considered to be a true positive for an object. Otherwise, it is considered to be a false positive. A false negative occurs when the model fails to detect an object that is present. Recall is the ratio of the number of true positives to the total number of actual objects in the image. Recall measures the accuracy of the model in predicting all the actual objects. Precision is the proportion of true positives to the total number of positive predictions. Precision increases with increasing true positives and decreases with increasing false positives. Below equations present the calculation of the precision and recall, respectively. Mean average precision (mAP) is the mean of average detected precision for all defect categories. Average precision (AP) is the area under the curve of precision versus recall.

$$Precision = \frac{True\ Positive}{True\ Positive + False\ Positive}$$

$$Recall = \frac{True\ Positive}{True\ Positive + False\ Negative}$$

3.3 Dataset

For this study, we used the benchmark dataset GC10-DET from the literature, which is available on GitHub. GC10-DET includes a steel surface defect dataset from real industry applications including punching hole, weld line, crescent gap, water spot, oil spot, silk spot, inclusion, rolled pit, crease, waist folding defect categories, which were recoded as Pu, Wl, Cg, Ws, Os, Ss, In, Rp, Cr, Wf, respectively (Lv et al., 2020). The total number of images in the dataset is 2306 and the size of each image is 2048 x 1000 pixels. The number of images in each defect category, as well as the train and test datasets used in this study are shown in Table 3-1.

Table 3-1 Number of Images per Category

Image Usage	Defect Category										Number of Images
	Pu	**Wl**	**Cg**	**Ws**	**Os**	**Ss**	**In**	**Rp**	**Cr**	**Wf**	
Training	176	223	201	235	169	531	177	26	47	127	1912
Testing	63	84	32	54	48	132	36	7	6	19	394
All Images	219	273	226	289	204	650	216	31	52	146	2306

3.4 Data augmentation

Image data augmentation is the process of artificially expanding the size of a dataset by creating modified versions of images in the dataset. In real time practice, it is not possible to include all the varieties of conditions that can impact the image quality in the dataset. Thus, data augmentation is important in these cases and all the augmentation techniques should be selected in such a way that they are closely related to the application/target goal. Considering this criterion, rotation, blur, and contrast suit best for this application. The performance of any deep neural network is greatly dependent on the number of images in the training dataset. In this study, since the number of images in the original dataset is small at least for some classes and it is evident that there is class imbalance with respect to some classes like Rp, data augmentation techniques were adopted to increase the dataset so that algorithm gets enough number of samples to get familiar with the defect classes. Data augmentation techniques considered in this study are rotation, gaussian blur, median blur, bilateral blur, box blur, contrast techniques like Min-Max contrast stretching, CLAHE (contrast limited adaptive histogram equalization) contrast stretching and histogram equalization method.

3.4.1 Rotation

The rotation technique changes the angle of the target object that appears in the dataset and this augmentation technique benefits applications where there is a chance that an image gets

skewed during the regular application. This technique can even help in the applications where the camera position is fixed relative to the object. The variations produced by this technique can help prevent the model from memorizing the data and help to combat potential overfitting. Below figure shows an example of the rotation technique.

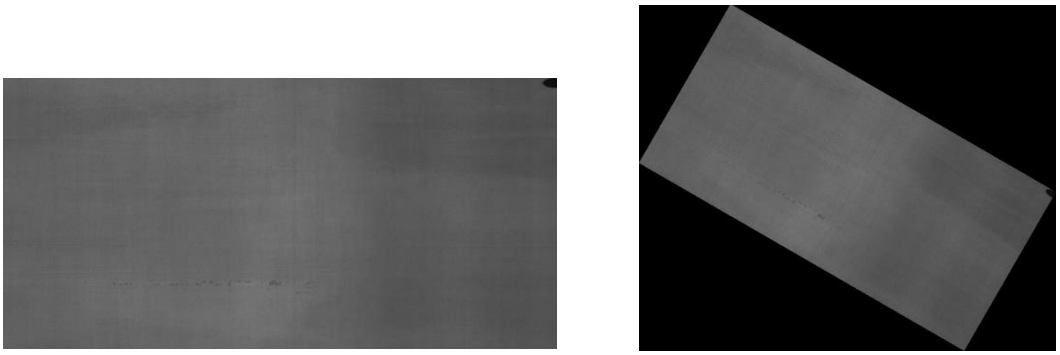

Figure 3-3 Original image (left) and Rotated image (right)

3.4.2 Blur techniques

Image blur is another important technique for data augmentation. These techniques generate blurred images which are used for training the algorithm. For the current study, four types of blur techniques are being evaluated. Each blur technique differs statistically from each other. These four techniques being used include median blur, bilateral blur, gaussian blur, and box blur. For all the blur operations a kernel with some specific shape (m x n) is used. This 2D filter (i.e., kernel such as the 3 x 3 shown in Fig 3-4) will be used to convolve with an image and statistical methods are used to make the modifications to the pixel values in the image. The type of statistical method determines the type of blur introduced.

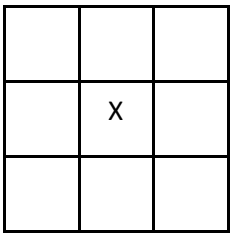

Figure 3-4 Kernel with size 3 x 3 and center of the kernel represented with 'X'

3.4.2.1 Median blur

In the median blur technique, a 2D filter is used to convolve with an image and median of all the pixel intensity values that lie in the kernel is calculated. This median value is designated as the new pixel intensity value to the pixel which is positioned at the center of the kernel. Here as its name indicates the statistical method 'median' is calculated to get the new pixel value. Below figure 3-5 shows the original image and median blur image with kernel 9 x 9.

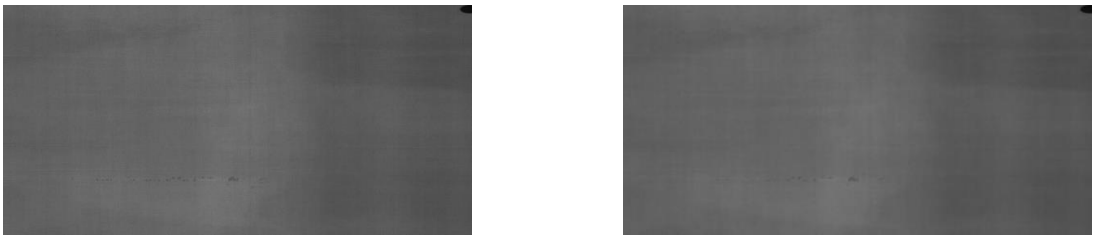

Figure 3-5 Original image (left) and median blur image with kernel 9 x 9 (right)

3.4.2.2 Gaussian blur

Gaussian blur uses the Gaussian kernel, which gives more weight to pixels closer to the center. In the application of Gaussian blur, a weighted average of the color values of the pixels in the kernel is calculated and because of the property of the Gaussian kernel, the pixels nearest to the center of the kernel are given more importance than those far away. Because of this, larger

kernels will blur the image more than smaller kernels. Figure 3-6 & 3-7 shows the example of the gaussian blur image along with their original image.

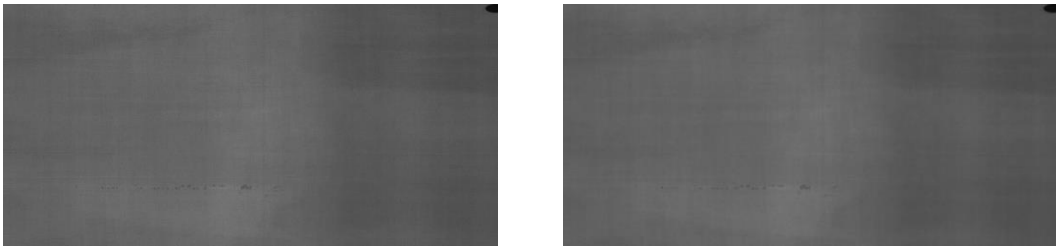

Figure 3-6 Original image (left) and Gaussian blur image with kernel 11 x 11 (right)

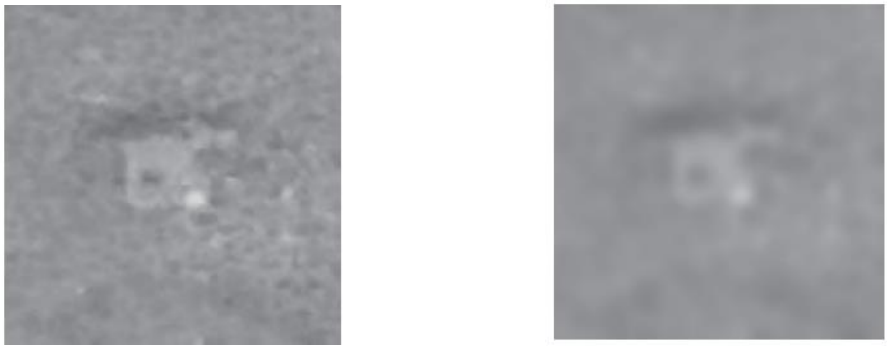

Figure 3-7 *Original image (left) and Gaussian blur image (right) (Adopted from Hoang, 2019)*

3.4.2.3 Bilateral blur

In the Gaussian filter, only nearby pixels are considered while filtering. It doesn't consider whether pixels have almost the same intensity or the pixel is an edge or not. Due to this, Gaussian blur affects the edges of the target object. Therefore, we implemented the bilateral filter blur to have some more variant blur images. Bilateral filter does not average the pixels near the edges rather it preserves the edges. Also, bilateral filter considers the pixels whether they have similar intensities. It takes account of the Gaussian blur (space parameter), and it also considers a

function of pixel intensity difference. So, it preserves the edges since pixels at edges will have large intensity variation. Figure 3-8 shows the example of the bilateral blur technique.

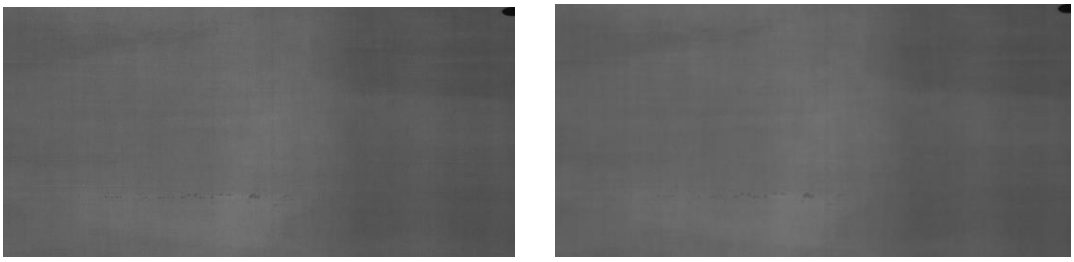

Figure 3-8 Original image (left) and Bilateral blur image with kernel 11 x 11 (right)

3.4.2.4 Box blur

The box blur is also known as the box linear filter (GeeksforGeeks, 2020). The blurred color of one pixel is the average of the pixel's color value and its neighboring pixels. Below figure shows the original images and its blurred image using box blur technique.

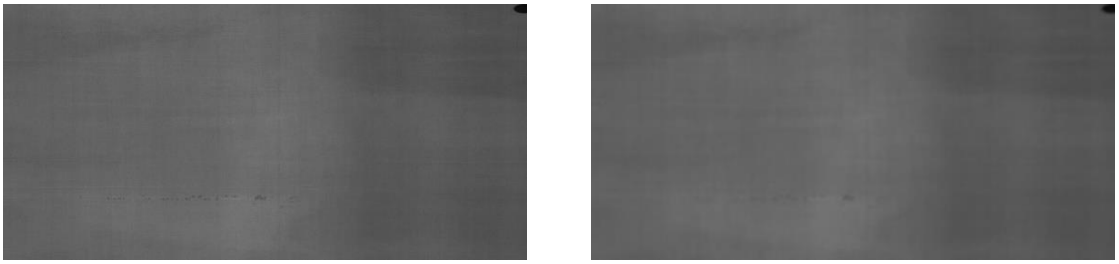

Figure 3-9 Original image (left) and box blur image with kernel 7 x 7 (right)

3.4.3 Contrast stretching techniques

3.4.3.1 Min-max contrast stretching

Min-max stretching is a linear stretching mechanism which uses the tails of the pixel intensity values and stretches the pixel values throughout the region. In Figure 3-10, pixel intensity values were stretched using the end points (r1, s1) and (r2, s2). Figure 3-11 shows the example of original, and contrast stretched image.

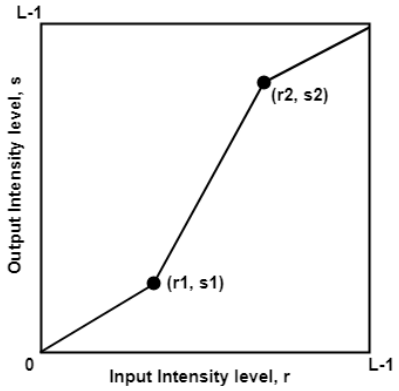

Figure 3-10 Linearly stretching the intensity values using the end points

Mathematically, min-max stretching for new pixel values is represented as:

$$X_{new} = \frac{X_{input} - X_{min}}{X_{max} - X_{min}} \times 255$$

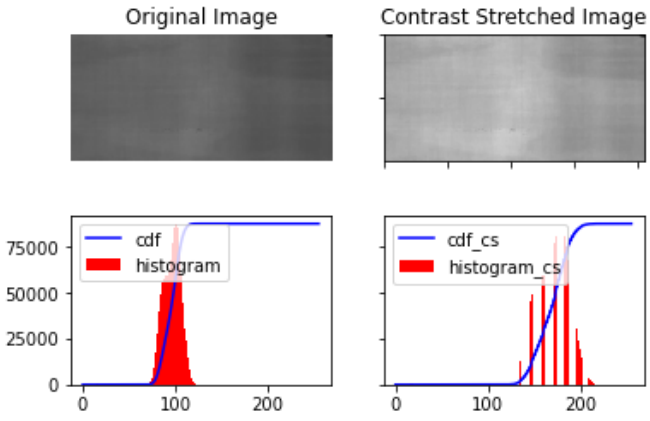

Figure 3-11 Linear stretching of the pixel values using min-max contrast technique.

3.4.3.2 Histogram Equalization

Histogram equalization is a non-linear contrast enhancement technique which uses non-linear functions to reassign the intensity values of an image. In contrast to the linear techniques, this technique does not use lower and higher intensity values for reassigning the new pixel values. Figure 3-12 shows the example of original and histogram equalized contrast image. The non-linear function shown in below equation is used to assign the new pixel value:

$$Sk = (L-1)\sum_{j=0}^{k} pr(rj)$$

where, $Pr(rj) = \frac{nj}{Total\ number\ of\ pixels}$ & n_j = number of pixels that have intensity rj

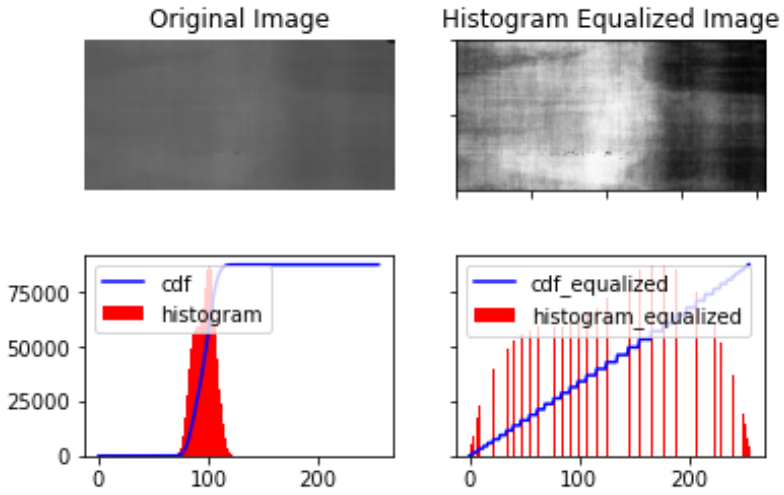

Figure 3-12 Linear stretching of the pixel values using histogram contrast technique.

3.4.3.3 Contrast Limited Adaptive Histogram Equalization (CLAHE)

Contrast limited adaptive histogram equalization (CLAHE) is also a non-linear method and improves the contrast of the image by stretching the intensity values using the technique called tiles. CLAHE is like histogram equalization; however, it operates on small regions in the image called tiles and all the neighboring tiles are then combined using bilinear

interpolation to remove the artificial boundaries. Figure 3-13 shows the example of original and CLAHE stretched image.

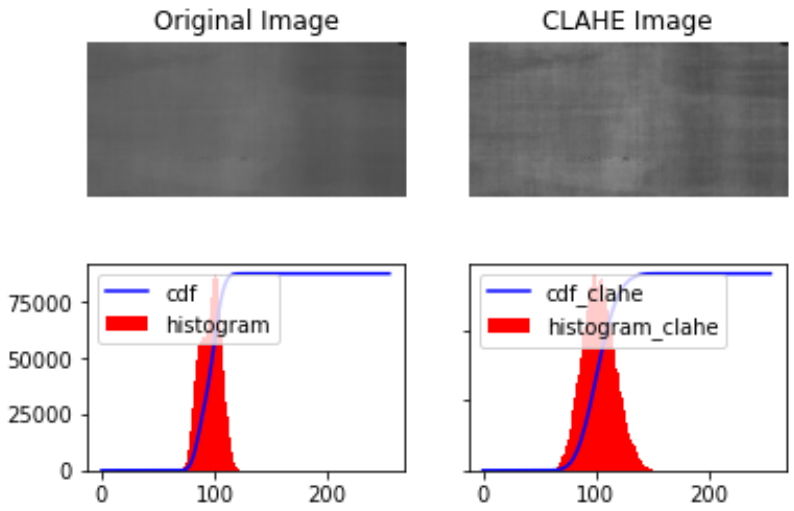

Figure 3-13 Linear stretching of the pixel values using CLAHE contrast technique.

3.4 Overview of Methodology

The fig 3-14 shows the overall methodology intended to use in the industry for defects detection.

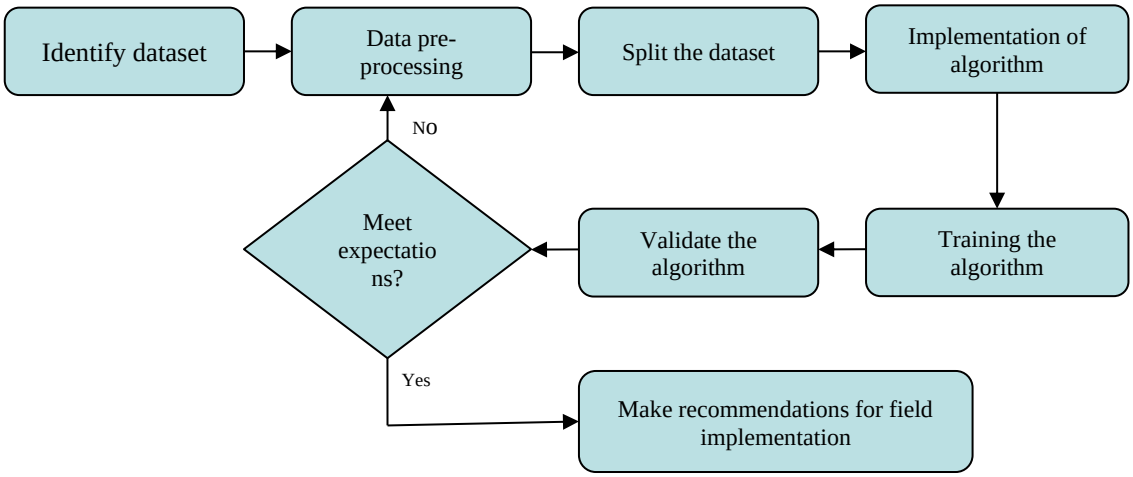

Figure 3-14 Overview of the methodology

3.5 Training

For training, the images were divided into several batches, and each batch was used to train the network each time. Batch normalization (Ioffe et al., 2015) was implemented in this network to normalize the data for each batch. Batch normalization stabilizes the learning process and reduces the number of epochs required to train the deep networks. It also helps with some regularization and reduces the generalization error. The YOLO network was trained with 20,000 training steps and the batch size and subdivisions were 64 and 16, respectively. The momentum and decay were selected as 0.949 and 0.0005. The learning rate was 0.0013. Leaky ReLu (Maas et al., 2013) and Mish (Misra et al., 2019) activation functions were used for the final and other layers.

Bag of Specials for detector includes the Leaky ReLu activation function, SPP-block, and Greedy NMS.

Bag of Freebies for detector includes the CIoU-loss function, mosaic data augmentation, elimination of grid sensitivity, and multiple anchors for single ground truth.

The loss function is given by (Zheng et al., 2020):

$$Loss_{CIoU} = \ 1 - IoU + \frac{|C - B \cup B^{gt}|}{C}$$

where C is the smallest box that covers the predicted bounding box B and ground truth (gt) bounding box B^{gt}.

CHAPTER 4. RESULTS

4.1 Results and discussion

This study used YOLO-V4 with Google Colaboratory (Bisong et al., 2019) that offers GPU ‘Tesla P100-PCIE' with a total memory of 16 GB. YOLO-V4 was pre-trained on both the ImageNet dataset and MS COCO dataset to extract basic image features such as edge and texture. Then, adopting the pre-trained model, YOLO network was trained for 20,000 iterations on ten types of defect images. The input size of 608 x 608 is used for training and testing.

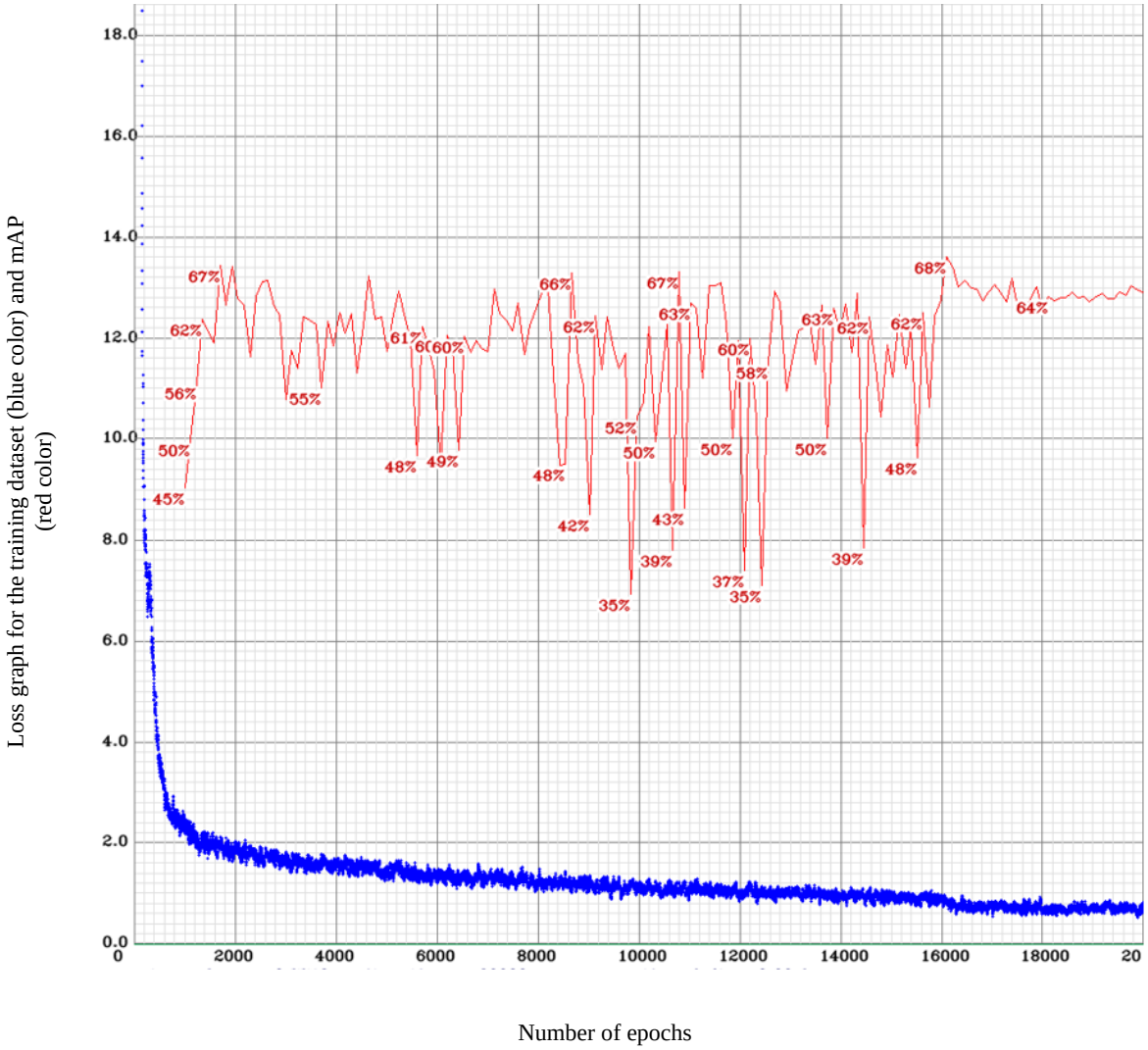

Figure 4-1 Loss decay curve during training and mAP values

The loss decay curve during training and mAP values obtained during testing using original dataset shown in Figure 4-1. We can see that loss values reduce gradually with fluctuations in the mAP values; mAP over ten classes are observed as 66.8% using the original dataset with no data augmentation. The mAP value using the augmented dataset is observed as 93.6%, the highest of all cases. The detection time per image is observed as 0.022 sec, which is faster and can be deployed in the production shop floor.

Although the test and training split of other methods (which plays a vital role in determining the algorithm accuracy) is unknown, Table 4-1, Table 4-2 shows the performance of the YOLO-V4 using both original and augmented dataset compared with other state-of-the-art algorithms, namely SSD, Faster R-CNN, YOLO-V2, YOLO-V3, and EDDN based on SSD as per the study (Lv et al. 2020). Table 4-1 shows the results for recall and Table 4-2 shows the results for average precision (AP) and mean average precision (mAP).

From Table 4-1, we can see that recall values of YOLOv4 with data augmentation technique performed better than all other methods for all the minority classes like Inclusion, rolled pit, crease and waist folding. Also, it performed better for the defect classes, oil spot and water spot classes where there is high similarity between the defects. For the defects like welding line the recall value is less because there are chances that algorithm gets confused between welding line and pale continuous line due to change in intensity portions in the image. Similar to recall values, average precision of YOLOv4 using augmented dataset performed better than YOLOv4 trained with no augmented dataset.

From Table 4-2, we can see that YOLOv4 with data augmentation technique outperformed all other methods for all the classes (except welding line defect). For welding line defect, the average precision is less compared with SSD because of the false positives error. The

algorithm predicting a pale continuous line as welding line defect and there are chances that it might be a labelling error since there are no standard rules mentioned by the author for labelling the GC10-DET dataset. Also, for the minority or ambiguous defect classes YOLOv4 trained with augmented dataset outperformed the other techniques by a huge margin. The impact of data augmentation on the performance of the object detection algorithm can be seen from results of YOLOv4 trained with non-augmented and with augmented dataset.

Table 4-1 Comparison of Recall values with respect to different methods (Adapted from Lv et al., 2020)

Recall							
Defect Category	**SSD**	**Faster R-CNN**	**YOLO-V2**	**YOLO-V3**	**EDDN based on SSD**	**YOLO-V4**	**YOLO -V4 with augmentation**
Pu	0.964	0.964	0.857	0.964	0.965	0.920	0.950
Wl	1.000	0.623	0.869	0.869	0.967	0.930	0.910
Cg	0.968	0.968	0.936	0.871	0.969	0.880	0.910
Ws	0.696	0.696	0.674	0.609	0.739	0.800	0.980
Os	0.848	0.761	0.63	0.565	0.891	0.610	0.970
Ss	0.956	0.708	0.694	0.542	0.988	0.530	0.760
In	0.578	0.551	0.444	0.311	0.667	0.360	0.860
Rp	0.667	0.333	0.333	0.333	0.333	0.330	1.000
Cr	0.571	1.000	0.429	0.429	0.857	0.830	1.000
Wf	1.000	0.800	0.900	0.700	1.000	0.790	1.000

Table 4-2 Comparison of AP and mAP values with respect to different methods (Adapted from Lv et al., 2020)

Average Precision (AP)							
Defect Category	**SSD**	**Faster R-CNN**	**YOLO-V2**	**YOLO-V3**	**EDDN based on SSD**	**YOLO-V4**	**YOLO -V4 with augmentation**
Pu	0.860	0.899	0.725	0.836	0.900	0.887	0.902
Wl	0.974	0.554	0.328	0.241	0.885	0.896	0.902
Cg	0.861	0.872	0.819	0.752	0.848	0.854	0.903
Ws	0.552	0.599	0.476	0.495	0.558	0.780	0.997
Os	0.612	0.653	0.403	0.329	0.622	0.617	0.905
Ss	0.689	0.579	0.473	0.325	0.650	0.456	0.873
In	0.168	0.194	0.096	0.036	0.256	0.363	0.881
Rp	0.105	0.364	0.018	0.036	0.364	0.295	1.000
Cr	0.527	0.736	0.212	0.429	0.521	0.773	1.000
Wf	1.000	0.818	0.614	0.400	0.919	0.717	1.000
mAP	0.635	0.627	0.433	0.388	0.651	0.668	0.936

Figure 4-2(a): Test images: Pu, Wl, Cg, Os, Ws defect classes

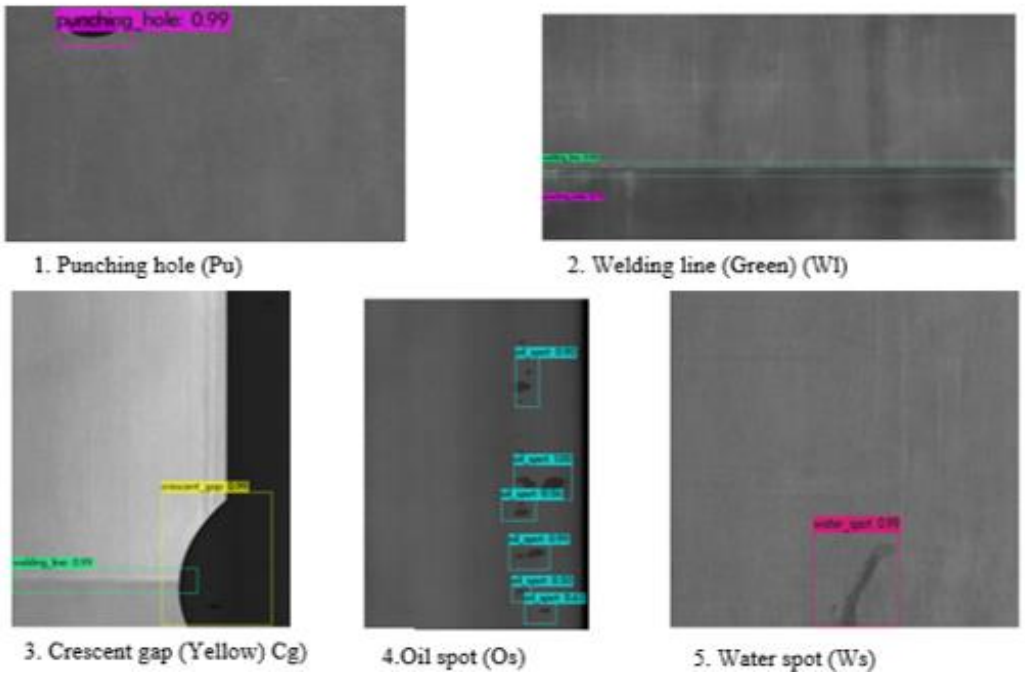

Figure 4-2(a) shows the test images with bounding box detections for the defect classes such as Punching hole, Welding line, Crescent gap, Oil spot, Water spot. Figure 4-3(b) shows the bounding box detections for the remaining defect classes such as Silk spot, Inclusions, Rolled pit, Crease, Waist folding.

Figure 4-2(b): Test images: Ss, In, Rp, Cr, Wf defect classes

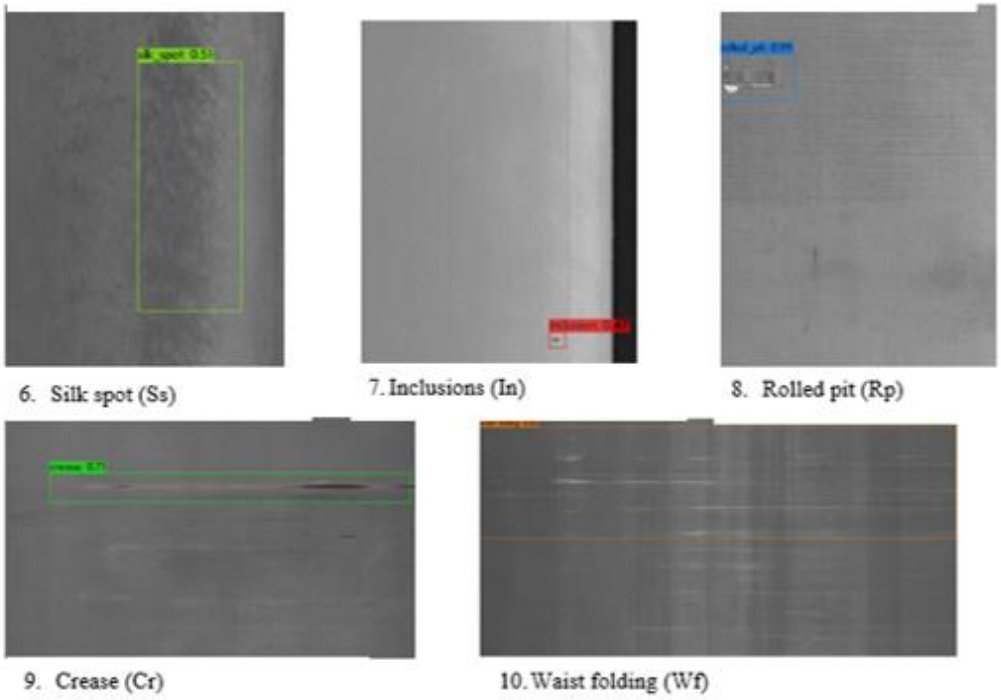

Figure 4-2 (Continued)

4.2 Insights from Experiments

It is common knowledge that training the machine learning model with very little data gives poor results. The size of the training dataset plays a crucial role on the performance of the model. Too little data can result in an optimistic and high variance estimation of model performance. Also, small dataset may cause the chosen machine learning model overfits the training set due to insufficient representation of the problem and will result in low-test accuracy. On the other hand, too much of data may cause the model to perform lower than the ideal test accuracy because of the difficulty to learn nuance of such large training dataset or sometimes due to over-representation of the problem.

Machine learning algorithms requires large amounts of data to give better results. For example, neural networks are the ones that requires copious amounts of training data. If the

architecture is deep, then the model needs more data to produce viable results. Reusing data is not a good idea, and data augmentation is recommended incase left with no choice of collecting more data due to complexity in data collection.

Two metrics plays a crucial role when it comes to the size of a dataset. They are 1. Bias and 2. Variance. Bias is expressed as the difference between observed value and the predicted value. For the models with high bias, models possess underfitting. Variance is defined as the difference in performance on the training set vs on the test set. The major issue with high variance is the model fits the training data well, but it does not generalize well on out of training datasets. This is the reason behind the use of validation and test set in the model building process. Data augmentation is used in this study to increase the dataset without overfitting and to help the model to learn the pattern of the classes in the images. To check the bias in the model, experiment is conducted ten iterations using the stratified sampling technique with replacement. Below results from the table 4-3 and 4-4 shows that results were consistent for both original and augmented dataset. It indicates that model performance is reliable.

Table 4-3 mAP values for ten iterations to validate the performance of the model for bias using original dataset

Test iterations	1	2	3	4	5	6	7	8	9	10
mAP	88.14%	87.64%	90.84%	88.37%	88.13%	90.48%	90.01%	88.64%	89.25%	89.41%

Table 4-4 mAP values for ten iterations to validate the performance of the model for bias using Augmented dataset

Test iterations	1	2	3	4	5	6	7	8	9	10
mAP	92.40%	92.48%	93.69%	92.22%	93.18%	92.52%	92.98%	93.46%	92.19%	91.93%

The advantage of the YOLOv4 algorithm is that it is a one stage object detector and have a glance on the image and produces the results with small amount of time. Along with its computationally excellence, this method can be easily trained and deployed in the production system. This requires only one GPU and best to operate in the real time case studies. In the current study, it just took 0.022 sec per image to produce the detection results with their labels.

The performance of the model on the given dataset depends on two factors: number of images and ambiguity. If the number of training images is less, then model will not get opportunity to learn the pattern of the defects and performs poor. Also, if the similarity between two defects is same then model needs large number of images to overcome the ambiguity between the classes.

Table 4-5 Type I and Type II errors of each defect classes using the original and augmented dataset

	Original dataset (Avg IoU - 53.65)					Augmented dataset (Avg IoU - 73.31%)				
Class	TP	FP	FN	Recall	Average Precision	TP	FP	FN	Recall	Average Precision
Pu	58	12	5	0.92	88.68%	60	8	3	0.95	90.21%
Wl	79	5	6	0.93	89.63%	77	7	8	0.91	90.24%
Cg	28	5	4	0.88	85.41%	29	3	3	0.91	90.32%
Ws	45	18	11	0.8	78.01%	55	2	1	0.98	99.69%
Os	67	24	43	0.61	61.72%	107	5	3	0.97	90.46%
Ss	86	43	76	0.53	49.56%	123	11	39	0.76	87.27%
In	25	24	44	0.36	36.33%	59	8	10	0.86	88.12%
Rp	6	7	12	0.33	29.47%	18	2	0	1	100.00%
Cr	5	1	1	0.83	77.27%	6	0	0	1	100.00%
Wf	15	7	4	0.79	71.75%	19	0	0	1	100.00%
					66.78%					93.63%

From the above table 4-5, we can see that false positives and false negatives were high for the classes Os, Ss, and In. Due to similarity between the defect classes with respect to their size and shape, the model failed due to ambiguity. Some of the Ss and In defects were detected as Ws. Inclusions (In) defects were detected as Os and Ws, Ss, Os were detected as In class. The

main reason for this poor performance is due to similarity between the classes. But with the help of augmented dataset the performance is enhanced in terms of false positives and false negatives. Though the number of false negatives were still high for Ss even with more number of images but the results were better in comparison with other methods.

Example for Silk spot defect false positives:

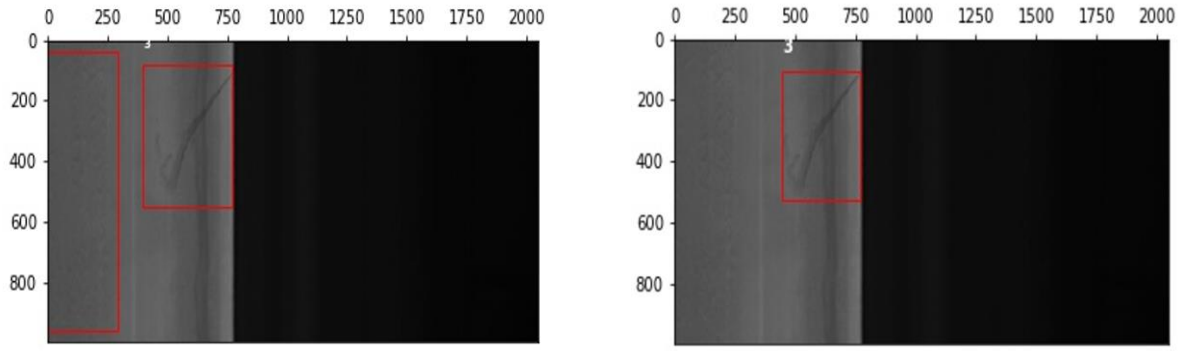

Figure 4-4 Detected image with label (left) and original image with ground truth label (right)

In the figure 4-3, there is no Silk spot defect, but model predicted some foreign feature as defect and represents the false positives. In this similar way 46 images were detected as false positive for the defect silk spot.

Example for Inclusion defect false positive:

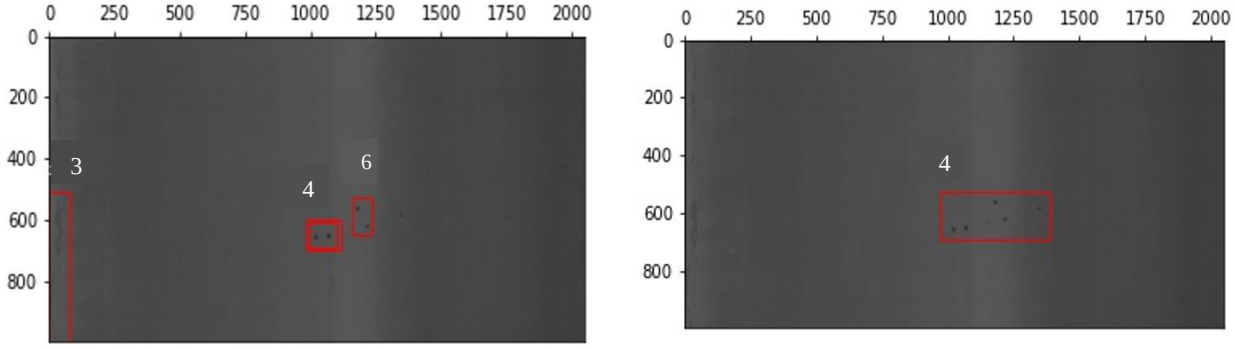

Figure 4-5 Detected image with label (left) and original image with ground truth label (right)

In the figure 4-4, oil spot defects were misclassified as Inclusion defect due to their close resemblance. This is one of the reasons for the Inclusion defects having false positives.

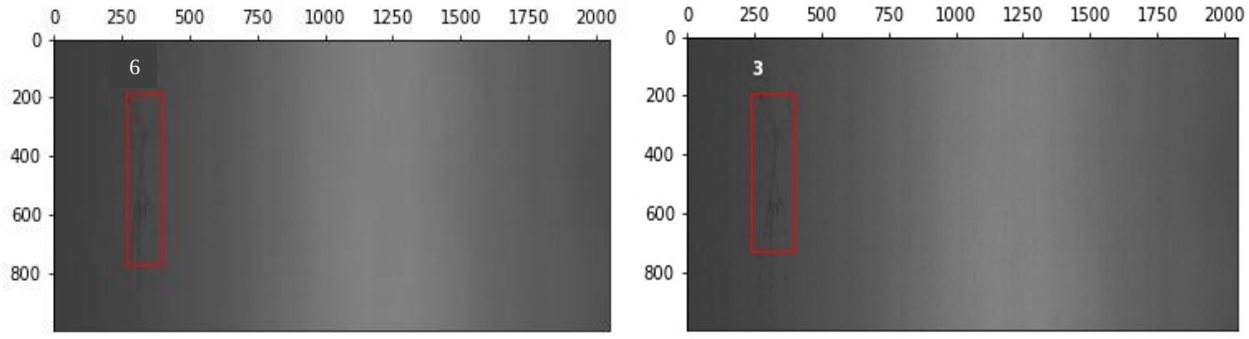

Figure 4-6 Detected image with label (left) and original image with ground truth label (right)

In the figure 4-5, water spot defects were misclassified as Inclusion defect due to their close resemblance in terms of their size and shape. This is one of the reasons for the Inclusion defects having false positives.

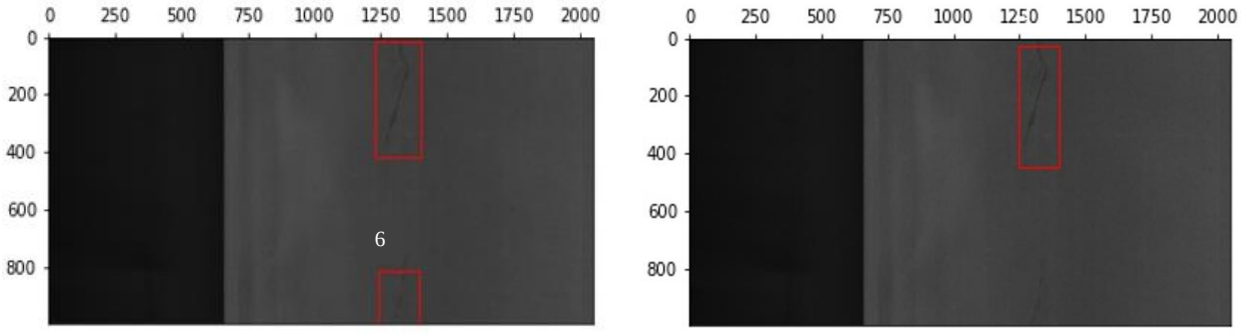

Figure 4-7 Detected image with label (left) and original image with ground truth label (right)

In the figure 4-6, there is no Inclusion defect, but model predicted some foreign feature as defect and represents the false positives. From all the three ways explained above, 24 images were detected as false positive for the defect Inclusion.

Overall, the main reason for false positives were detecting some foreign objects as defects or incorrect detection of other classes as shown in the above examples.

Defect size and shape played crucial role in this study. The average precision is higher for the defects like Punching hole, welding line and crescent gap due to their distinctive shapes and large sizes when compared to the other defects. For the other defect classes, size is small, and model required more images to get better results with the help of data augmentation. For Machine learning/deep learning model, size of the dataset depends on the complexity and size of the defects, bias, variance, and type of algorithms like transfer learning used for the study. If the target size is large and has distinct features, then model requires less amount of dataset to learn the pattern of the features.

False positives and False negatives are another important metrics considered for this study. False positives represent the Type I and false negatives represent the Type II errors. In this metal defect case study, false negatives are not acceptable because skipping the defects for rework can cause the defective product delivering to the customer. False positives cause the investment of extra time for inspecting the component and it is kind of acceptable if the false positives are less for any class, but false negatives are strictly not allowed as this cause the loss of customer's trust and reliability.

4.3 Limitations and Future Work

There is still scope for improvement of the precision values in terms of FNs and FPs. The main limitation of this work is to get the required quantity and quality of the dataset. To increase the size of the dataset, degenerative networks like GANs can be used. GANs are the techniques which generates the images artificially using the original images. Study can be conducted using GANs and investigation can be done for better results. Also, an attempt can be made with

unsupervised learning method. The advantage of the unsupervised method is that there is no need of labelling the defect classes and this saves lot of time for preparing the data. Also, it eliminates the errors involved in the labels of the dataset. One incorrect label can impact the performance of the model significantly. Unsupervised learning method first finds the unknown patterns in the data and these pattern or features were used to cluster the classes as groups. Also, it is good idea to use the semi supervised model if there are any missing labels. This work can be extended by implementing the methods which can achieve better accuracy. Ablation study can be conducted to study the inner representation of the network. Ablation study determine the importance of specific parts of the network that are crucial for the detection process. Also, ablation study helps to identify the redundant representations in specific parts of the network that damages the performance of the network.

CHAPTER 5. CONCLUSIONS

The process of assessing the condition and cleaning of candidate parts is an important step in remanufacturing. Traditionally, the inspection process in remanufacturing is carried out manually, which is a tedious process that is prone to human error. The goal of the inspection process is to locate and recognize the defects. There is always a chance that manual inspections may fail to locate the defects and as a result parts that cannot be remanufactured are released to remanufacturing processes, incurring unnecessary costs. To overcome this problem and to improve user confidence, this study has explored the capabilities of an object detection method to characterize defects in images of steel surfaces.

We implemented the YOLO-V4 object detection model and evaluated its effectiveness using an existing defect image dataset, GC10-DET. The dataset is very challenging due to unbalanced image sizes and similarity between different defect classes. Data augmentation techniques were implemented to increase the number of images per class available for training the algorithm. With this increase in dataset, model achieved familiarity with the defects and thereby reduced ambiguity due to similar classes in the data.

Despite challenges with the dataset, the performance of YOLO-v4 with no augmentation in terms of recall, AP, and mAP is comparable with other state-of-the-art techniques. Using data augmentation techniques, YOLOv4 showed its superiority over other techniques. The performance of the YOLOv4 when using no augmentation techniques is less because of higher false positives. False positives can occur because of two reasons. Incorrect detection of non-existing object or misplaced detection of an existing object. In the first case with no detection, it is observed that for some images, defects of water spot and oil spot are detected as silk spot or inclusion. Some of the inclusion defects were detected as water spot and small semi-circular

holes were detected as punching holes. Because of these false positives, the performance lacked behind and with the help of data augmentation, the performance of the algorithm improved greatly.

The selection of data augmentation techniques is completely dependent on the application, indicating that there is a need for the augmentation techniques that can be implemented in any application. With the help of other available Bag of Specials and Bag of Freebies techniques, the features may be detected more accurately without using augmentation techniques to save time with less false positives and false negatives.

As provided in the previous chapter, YOLOv4 algorithm with augmented dataset outperforms all other techniques in most of the cases. The recall values for some of the defects are less in comparison with other techniques; this is because of two reasons: due to labelling error in the original dataset and due to algorithm's performance issue. Algorithm's performance issue is mainly due to similarity between the defect classes. The standards for labelling the GC10DET is not mentioned and there is a confusion with some of the defects like weld line.

REFERENCES

Baumgartl, H., Tomas, J., Buettner, R., & Merkel, M. (2020). A deep learning-based model for defect detection in laser-powder bed fusion using in-situ thermographic monitoring. *Progress in Additive Manufacturing*, 1–9.

Bisong, E. (2019). Google Colaboratory. In *Building Machine Learning and Deep Learning Models on Google Cloud Platform: A Comprehensive Guide for Beginners* (pp. 59–64). Apress. https://doi.org/10.1007/978-1-4842-4470-8_7

Bochkovskiy, A., Wang, C.-Y., & Liao, H.-Y. M. (2020). Yolov4: Optimal speed and accuracy of object detection. *ArXiv Preprint ArXiv:2004.10934*.

Cha, Y.-J., Choi, W., & Buyukozturk, O. (2017). Deep Learning-Based Crack Damage Detection Using Convolutional Neural Networks. *Computer-Aided Civil and Infrastructure Engineering*, *32*, 361–378. https://doi.org/10.1111/mice.12263

Cho, J., Lee, K., Shin, E., Choy, G., & Do, S. (2015). How much data is needed to train a medical image deep learning system to achieve necessary high accuracy? *ArXiv Preprint ArXiv:1511.06348*.

DeVries, T., & Taylor, G. W. (2017). Improved regularization of convolutional neural networks with cutout. *ArXiv Preprint ArXiv:1708.04552*.

Errington, M., & Childe, S. J. (2013). A business process model of inspection in remanufacturing. *Journal of Remanufacturing*, *3*(1), 1–22.

Essid, O., Laga, H., & Samir, C. (2018). Automatic detection and classification of manufacturing defects in metal boxes using deep neural networks. *PloS One*, *13*(11), e0203192.

Gai, X., Ye, P., Wang, J., & Wang, B. (2020). Research on defect detection method for steel metal surface based on deep Learning. *2020 IEEE 5th Information Technology and Mechatronics Engineering Conference (ITOEC)*, 637–641.

GC10-DET Metallic Surface Defect Datasets, Github, Available: https://github.com/lvxiaoming2019/GC10-DET-Metallic-Surface-Defect-Datasets, Retrieved 25 March 2021

GeeksforGeeks. 2020. QuickSort-GeeksforGeeks. (online) Available at: <https://www.geeksforgeeks.org/box-blur-algorithm-with-python-implementation/> (Accessed 30 December 2020).

Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 580–587.

Guide, J. V. D. R., & Van Wassenhove, L. N. (2002). The reverse supply chain. *Harvard Business Review*, *80*(2), 25–26.

Hammond, R., Amezquita, T., & Bras, B. (1998). Issues in the automotive parts remanufacturing industry: a discussion of results from surveys performed among remanufacturers. *Engineering Design and Automation*, *4*, 27–46.

He, K., Zhang, X., Ren, S., & Sun, J. (2015). Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *37*(9), 1904–1916.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 770–778.

Hoang, N.-D. (2018). An artificial intelligence method for asphalt pavement pothole detection using least squares support vector machine and neural network with steerable filter-based feature extraction. *Advances in Civil Engineering*, *2018*.

Huang, G., Liu, Z., Van Der Maaten, L., & Weinberger, K. Q. (2017). Densely connected convolutional networks. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 4700–4708.

Ijomah, W. L., McMahon, C. A., Hammond, G. P., & Newman, S. T. (2007). Development of robust design-for-remanufacturing guidelines to further the aims of sustainable development. *International Journal of Production Research*, *45*(18–19), 4513–4536.

Intel. (2018). *IT@INTEL Faster, More Accurate Defect Classification Using Machine Vision* [White paper]. Intel. https://www.intel.com/content/dam/www/public/us/en/documents/best-practices/faster-more-accurate-defect-classification-using-machine-vision-paper.pdf

Ioffe, S., & Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. *International Conference on Machine Learning*, 448–456.

Joulin, A., Maaten, L. van der, Jabri, A., & Vasilache, N. (2016). Learning visual features from large weakly supervised data. *European Conference on Computer Vision*, 67–84.

Kandukuri, S. (2019). *TRIZ inspired design guidelines for remanufacturing using additive manufacturing*. Iowa State University.

Kopardekar, P., Mital, A., & Anand, S. (1993). Manual, hybrid and automated inspection literature and current research. *Integrated Manufacturing Systems*.

Lei, S., Zhang, H., Wang, K., & Su, Z. (2018). *How training data affect the accuracy and robustness of neural networks for image classification*.

Lin, H.-I., & Wibowo, F. S. (2021). Image Data Assessment Approach for Deep Learning-Based Metal Surface Defect-Detection Systems. *IEEE Access*, *9*, 47621–47638.

Lv, X., Duan, F., Jiang, J.-J., Fu, X., & Gan, L. (2020). Deep Metallic Surface Defect Detection: The New Benchmark and Detection Network. *Sensors (Basel, Switzerland)*, *20*(6). https://doi.org/10.3390/s20061562

Maas, A. L., Hannun, A. Y., & Ng, A. Y. (2013). Rectifier nonlinearities improve neural network acoustic models. Proc. Icml, 30(1), 3.

Masci, J., Meier, U., Fricout, G., & Schmidhuber, J. (2013). Multi-scale pyramidal pooling network for generic steel defect classification. *The 2013 International Joint Conference on Neural Networks (IJCNN)*, 1–8.

Misra, D. (2019). Mish: A self-regularized non-monotonic activation function. *ArXiv Preprint ArXiv:1908.08681*.

Mista, T. (2019, April 22). How do you know you have enough training data? Towards Data Science. https://towardsdatascience.com/how-do-you-know-you-have-enough-training-data-ad9b1fd679ee

Mital, A., Govindaraju, M., & Subramani, B. (1998). A comparison between manual and hybrid methods in parts inspection. *Integrated Manufacturing Systems*.

Montgomery, D. C. (1991). *Introduction to Statistical Quality Control: By Douglas C. Montgomery*. John wiley & Sons.

Natarajan, V., Hung, T.-Y., Vaikundam, S., & Chia, L.-T. (2017). Convolutional networks for voting-based anomaly classification in metal surface inspection. *2017 IEEE International Conference on Industrial Technology (ICIT)*, 986–991.

Olteanu, A. (2018, January 3). Tutorial: Learning Curves for Machine Learning in Python *Dataquest*. https://www.dataquest.io/blog/learning-curves-machine-learning/

Ortegon, K., Nies, L. F., & Sutherland, J. W. (2013). Preparing for end of service life of wind turbines. *Journal of Cleaner Production*, *39*, 191–199.

Rahaman, G. M. A., & Hossain, M. M. (2009). Automatic Defect Detection and Classification Technique from Image: {A} Special Case Using Ceramic Tiles. *CoRR*, *abs/0906.3*. http://arxiv.org/abs/0906.3770

Ren, R., Hung, T., & Tan, K. C. (2017). A generic deep-learning-based approach for automated surface inspection. *IEEE Transactions on Cybernetics*, *48*(3), 929–940.

Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in Neural Information Processing Systems*, *28*, 91–99.

Steinhilper, R., "Remanufacturing: The Ultimate Form of Recycling," Stuttgart: Fraunhofer IRB Verlag, 1998.

Sun, C., Shrivastava, A., Singh, S., & Gupta, A. (2017). Revisiting unreasonable effectiveness of data in deep learning era. *Proceedings of the IEEE International Conference on Computer Vision*, 843–852.

Sutherland, J. W., Jenkins, T. L., & Haapala, K. R. (2010). Development of a cost model and its application in determining optimal size of a diesel engine remanufacturing facility. *CIRP Annals*, *59*(1), 49–52.

Tao, X., Zhang, D., Ma, W., Liu, X., & Xu, D. (2018). Automatic metallic surface defect detection and recognition with convolutional neural networks. *Applied Sciences*, *8*(9), 1575.

Uijlings, J. R. R., Van De Sande, K. E. A., Gevers, T., & Smeulders, A. W. M. (2013). Selective search for object recognition. *International Journal of Computer Vision*, *104*(2), 154–171.

Wang, C.-Y., Liao, H.-Y. M., Wu, Y.-H., Chen, P.-Y., Hsieh, J.-W., & Yeh, I.-H. (2020). CSPNet: A new backbone that can enhance learning capability of CNN. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 390–391.

Warden, P. (2017, December 14). How many images do you need to train a neural network? *Pete Warden's Blog.* https://petewarden.com/2017/12/14/how-many-images-do-you-need-to-train-a-neural-network/

Yeum, C. M., & Dyke, S. J. (2015). Vision-based automated crack detection for bridge inspection. *Computer-Aided Civil and Infrastructure Engineering*, *30*(10), 759–770.

Yun, J. P., Shin, W. C., Koo, G., Kim, M. S., Lee, C., & Lee, S. J. (2020). Automated defect inspection system for metal surfaces based on deep learning and data augmentation. *Journal of Manufacturing Systems*, *55*, 317–324.

Zhang, H., Cisse, M., Dauphin, Y. N., & Lopez-Paz, D. (2017). mixup: Beyond empirical risk minimization. *ArXiv Preprint ArXiv:1710.09412*.

Zhao, Z.-Q., Zheng, P., Xu, S., & Wu, X. (2019). Object detection with deep learning: A review. *IEEE Transactions on Neural Networks and Learning Systems*, *30*(11), 3212–3232.

Zheng, Z., Wang, P., Liu, W., Li, J., Ye, R., & Ren, D. (2020). Distance-IoU loss: Faster and better learning for bounding box regression. *Proceedings of the AAAI Conference on Artificial Intelligence*, *34*(07), 12993–13000.

Zhong, Z., Zheng, L., Kang, G., Li, S., & Yang, Y. (2020). Random erasing data augmentation. *Proceedings of the AAAI Conference on Artificial Intelligence*, *34*(07), 13001–13008.