FIRAT UNIVERSITY
GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
TÜRKİYE

A HIGH-PERFORMANCE CONVOLUTIONAL NEURAL NETWORK FOR STEEL DEFECTS DETECTION

Stephen Ikechukwu OKO-EGWU

Master's Thesis

DEPARTMENT OF COMPUTER ENGINEERING
Program of
Computer Engineering

APRIL 2022

FIRAT UNIVERSITY
GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
TÜRKİYE

Department of Computer Engineering

Master's Thesis

A HIGH-PERFORMANCE CONVOLUTIONAL NEURAL NETWORK FOR STEEL DEFECTS DETECTION

Author
Stephen Ikechukwu OKO-EGWU

Supervisor
Prof. Dr. Erhan AKIN

APRIL 2022
ELAZIG

FIRAT UNIVERSITY
GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

TÜRKİYE

Department of Computer Engineering

Master's Thesis

Title:	A High-Performance Convolutional Neural Network for Steel Defects Detection
Author:	Stephen Ikechukwu OKO-EGWU
Submission Date:	14 January 2022
Defense Date:	01 April 2022

THESIS APPROVAL

This thesis, which was prepared according to the thesis writing rules of the Graduate School of Natural and Applied Sciences, Fırat University, was evaluated by the committee members who have signed the following signatures and was unanimously approved after the defense exam made open to the academic audience.

Signature

[redacted]

This thesis was approved by the Administrative Board of the Graduate School on

..... / ..... / 20 .....

Signature

Prof. Dr. Kürşat Esat ALYAMAÇ
Director of the Graduate School

DECLARATION

I hereby declare that I wrote this Master's Thesis titled “A High-Performance Convolutional Neural Network for Steel Defects Detection” in consistent with the thesis writing guide of the Graduate School of Natural and Applied Sciences, Firat University. I also declare that all information in it is correct, that I acted according to scientific ethics in producing and presenting the findings, cited all the references I used, express all institutions or organizations or persons who supported the thesis financially. I have never used the data and information I provide here in order to get a degree in any way.

01 April 2022

Stephen Ikechukwu OKO-EGWU

PREFACE

In the steel terminal manufacturing industry, defects on steel terminals during production are a great source of loss, both in revenue and in the general acceptability of products by end-users. This imposes the burden of designing and managing systems for detecting defects on steel terminal manufacturers. Also, deploying such defects detection systems that can work in near real-time, with robustness, and low compute cost is another important challenge. This forms the basis of this thesis, and the convolutional neural network designed for defect detection on steel terminals is based on the YOLO v3 model. However, the thesis faced the difficulties of large-scale industrial data collection as the author clearly lacks the capacity for such undertaking requiring many professionals working over several months, and also is limited in scope in that it is an experimental research work and not an enterprise-grade deployment.

This thesis was partly supported financially by the Scientific and Technological Research Council of Turkey (TUBITAK) in terms of equipment. Hatko Electronics Istanbul provided logistics including travel for the author, while also making their steel terminals production floor open to the author for data collection. Dr. Ahmet Topkaya, the R&D and Business Development Director at Hatko Electronics provided technical support with his R&D team in ensuring the author had access to the right tools for data acquisition.

This thesis was supported by the project number **MF.21.55** by Firat University Scientific Research Projects Coordination Unit (FÜBAP).

Stephen Ikechukwu OKO-EGWU
ELAZIG, 2022

TABLE OF CONTENTS

Page

ABSTRACT

A High-Performance Convolutional Neural Network for Steel Defects Detection

Stephen Ikechukwu OKO-EGWU

Master's Thesis

FIRAT UNIVERSITY
Graduate School of Natural and Applied Sciences

Department of Computer Engineering

April 2022, Page: xiii + 52

Ensuring defect-free products is a critical consideration in the hot-rolled steel strips and steel terminal manufacturing industry and cannot be over-emphasized, which underpins the purpose of the study presented herein which is focused on developing an algorithm for the detection of defects for two (2) different classes of steel products.

The study is divided into two (2) parts, part one being based on transfer learning is aimed at detecting defects on hot rolled steel strip surfaces, while part two aims to achieve defect detection on steel terminals using an optimized YOLO v3 algorithm.

Problem statement one of the study proposes a transfer learning-based method for detecting defects on hot-rolled steel strip surfaces. The method proposed in this study utilizes Deep Convolutional Neural Network (DCNN) using VGG-19 as a pre-trained model, with data augmentation to mitigate the effect of limited training data and also data-class imbalance. Experimental tests indicate that the method proposed herein using VGG-19 pre-trained model has an excellent performance, with training and validation accuracies of 93.11% and 97.22%, respectively, which is more accurate than the KNN, ANN, and Faster RCNN model.

Problem statement two of the study proposes an optimized YOLO v3 algorithm to detect three (3) most common classes of defects on steel terminals. The optimized YOLO v3 network proposed in the thesis was trained using steel terminal image datasets acquired in real-time from Hatko Electronics' production factory in Istanbul, and delivered a good performance in the region of 97.19% AP50, at 0.50 IOU Threshold, which is higher than the mAP (mean Average Precision) results (57.9% AP50 @0.50) from the original YOLO v3 network, and higher than the mAP score of Faster-RCNN that is about 55.7% AP50, and, 50.4% AP50 for SSD513 (Single-shot Detector) network.

Keywords: YOLO v3, Faster-RCNN, Defect Detection, Mean Average Precision, Single-Shot Detector, Computer Vision.

ÖZET

Çelik Kusurlarının Tespiti için Yüksek Performanslı Evrişimli Sinir Ağı

Stephen Ikechukwu OKO-EGWU

Yüksek Lisans Tezi

FIRAT ÜNİVERSİTESİ
Fen Bilimleri Enstitüsü

Bilgisayar Mühendisliği Anabilim Dalı

Nisan 2022, Sayfa: xiii + 52

Sıcak haddelenmiş çelik şeritlerde ve çelik terminal imalat endüstrisinde hatasız ürünler sağlamanın kritik ilgisi aşırı vurgulanamaz ve bu, iki (2) farklı çelik ürün sınıfı için bir kusur tespit algoritması geliştirmeye odaklanan bu çalışmanın amacının temelini atmaktadır.

Çalışma iki (2) parçaya ayrılmıştır ve birinci bölüm, sıcak haddelenmiş çelik şeritlerdeki yüzey kusurlarını tespit etmeyi öğrenme üzerine kuruludur. ikinci bölüm ise çelik terminallerdeki kusurları tespit etmek için bu çalışmada optimize edilen YOLO v3 algoritmasına dayanmaktadır.

Çalışmanın birinci bölümü, sıcak haddelenmiş çelik şeritlerdeki yüzey kusurlarını tespit etmek ve sınıflandırmak için transfer öğrenme tabanlı bir yöntem önermektedir. Önerilen yöntem, sınırlı sayıda eğitim verisi ve veri sınıfı dengesizliğinin etkisini azaltmak için veri büyütme ile önceden eğitilmiş bir VGG19 kullanarak aktarılan öğrenmeye dayanan bir Derin Eşlemsel Sinir Ağı (DCNN) modeli kullanmaktadır. Deneysel testler, VGG19 önceden eğitilmiş modeli kullanarak önerilen yöntemin, KNN, ANN ve Faster RCNN modelinden daha doğru olan sırasıyla %93.11 ve %97.22 eğitim ve doğrulama doğrulukları ile mükemmel bir performansa sahip olduğunu göstermektedir.

Çalışmanın ikinci bölümü, çelik terminallerdeki en yaygın üç (3) kusur sınıfını tespit etmek için optimize edilmiş bir YOLO v3 algoritması önermektedir. Tezde önerilen optimize edilmiş YOLO v3 ağı, Hatko Electronics'in İstanbul'daki üretim fabrikasından gerçek zamanlı olarak edinilen çelik terminal görüntü veri kümeleri kullanılarak eğitilmiştir, ve orijinal YOLO v3 ağından mAP (ortalama Ortalama Hassasiyet) sonuçlarından (%57,9 AP50 @0,50) daha yüksek olan 0,50 IOU Eşiği'nde %97,19 AP50 bölgesinde iyi bir performans sunmuş, ve yaklaşık %55.7 AP50 ve SSD513 (Tek Atış dedektörü) ağı için% 50.4 AP50 olan Faster-RCNN'nin mAP puanından daha yüksektir.

Anahtar Kelimeler: YOLO v3, Faster-RCNN, Defect Detection, Ortalama Ortalama Hassasiyet, Tek Atış Dedektörü, Bilgisayar Görüşü.

LIST OF FIGURES

Page

LIST OF TABLES

Page

LIST OF APPENDICES

Page

SYMBOLS AND ABBREVIATIONS

Symbols

σ: used in statistics to represent the standard deviation.

Abbreviations

ANN	: Artificial Neural Network
AP	: Average Precision
CVAT	: Computer Vision Annotation Tool
DCNN	: Deep Convolutional Neural Network
F-RCNN	: Fast Region-Based Convolutional Neural Network
GPU	: Graphical Processing Unit
IOU	: Intersection Over Union
VGG19:	Visual Graphics Group 19
KNN	: K-Nearest Neighbors
mAP	: Mean Average Precision
NEU	: Northeastern University
R-CNN	: Region-Based Convolutional Neural Network
RELU	: Rectified Linear Unit
S-CNN	: Siamese Convolutional Neural Network
SSD	: Single-Shot Detection
TX1	: Tegra version X1
YOLO	: You Only Look Once

1. INTRODUCTION

The relevance of quality control in an industrial production line can never be overemphasized and not be underestimated either. The importance of steel products of any form devoid of defects cannot be overemphasized and hence the need to develop robust defect detection algorithms to ensure high quality of steel products and its derivatives, increased revenue for steel manufacturers, as waste will be greatly curtailed while increasing end-user satisfaction.

There are loosely two categories of quality control mechanisms, namely; destructive or non-destructive. Non-destructive testing (NDT) is used to monitor a component for the sole purpose of detecting a defect without sampling or permanently damaging it [1]. Visual defect detection methods are one of the most widely used methods in the industry and a popular non-destructive approach.

Currently, machine learning developments, especially related to deep learning, are proving to be very helpful in detecting and classifying different types of flaws, mostly surface flaws in various industrial products such as steel. The culmination and significance of these advances is the unique capability of the deep learning methods and techniques to extract feature representations directly that are hierarchical from images, thereby eliminating the desire for hand-crafted features [2]. Deep learning has proven to be very interesting and promising in error detection, which has made it the darling of researchers worldwide.

Different models have been proposed by researchers in the Automated Optical Inspection (AOI) domain, to solve the problem of automating the industrial quality control and inspection process, in a manner that is not just accurate, but efficient as well, with minimal cost addition to the industries. Most importantly, the application of deep learning schemes like Deep Convolutional Neural Networks (DCNNs) has become a widespread. However, to train highly accurate and useful DCNNs, data is required in huge amounts (in the hundreds of thousands) to enable the network to learn important features (good generalization ability) and quickly converge on the training data. This has become a problem when applying DCNNs in industrial automated inspection tasks due to the lack of a significant amount of data sets [3].

To solve the small dataset problem, different approaches have been employed, among which are, data augmentation, transfer learning, hyper parameter tuning, among others. This work relies on data augmentation, transfer learning, and other cutting-edge approaches to design an optimized, high-performance, deep, convolutional neural network to detect surface defects in a way that is less computationally intensive and fast to converge even with limited data. The problem of training and detection time is very vital in industrial production environments, considering that efficient production processes are very important in industries, and any model to be deployed in industrial environments must be able to meet the time constraints of industrial machines. Again, compute cost

is another constraint for inspection models. This is considerably so, since many industrial facilities are pre-configured with ready-to-go industrial computing systems, without GPUs (mostly), and building models that can run on standard industrial computers is highly desirable, to ensure no extra costs are imposed on industries.

The authors in [4] have given an overview of the different approaches to industrial fault detection and classification. In their survey, they reviewed many automated visual flaw detection approaches, mainly applicable to different material types like metals, ceramics, and textiles. They identified two (2) major classes of defects, categorized as; visible (scratches, shape defects, etc.) and tangible (crack, dent, etc.). Recently, in [5], the authors developed a DCNN-based scheme meant for the detection and classification of various categories of defects that may occur during the process of laser welding when batteries are being manufactured. In addition, they threw up the VGG novel model in order to enhance efficiency in the classification of defects. When tested, their model achieved about 99.87% accuracy using over 8000 samples. Further, a S-CNN method for one-shot detection (identification) of defects on the surface of steel was propounded in [6]. This approach showed a significant reduction in the need for large data during training with an added advantage of the capability to be performed in real-time. The model achieved an accuracy of 92.55% on 80% of the NEU data set used in training. In [6] and [7] the authors recommended searching for better hyper-parameters for the NEU dataset they used in their work [8] and using high-channel, feature-rich sensor data. Also, they recommended using transfer learning for steel flaw detection to investigate the usefulness of pre-trained (VGG-19, Res-Net, [9] etc.) models for steel surface flaw detection. This provides the basis and motivation for part one of this study, aimed at examining the suitability or otherwise of the transfer learning approach in designing an algorithm primarily for the detection of defects on the surface of mostly hot-rolled steel strips with regards to accuracy and detection speed.

Furthermore, industrial domains such as automotive, electrical/electronic, shipbuilding, aerospace, aircraft, automobiles, machine tools, and machinery manufacturing, steel products play a very crucial and significant role [10]. However, defects are practically unavoidable in steel terminals, viz.; scratches, bent pin, and spot regions. These kind of defects do not just affect the appearance, dimensional specifications and suitability of the steel product, but also diminish their functional properties and impose huge economic losses [2]. Over the years, a manual inspection process has been deployed by steel terminal manufacturing industries with some having some automated system that is complex, costly, and less robust, and this has not proven to be satisfactorily adequate in guaranteeing defect-free steel terminals with a reasonable degree of reliability and fit-for-purpose. This inadvertently led to a rise in the development and deployment of different automatic surface defect detection mechanisms aimed at meeting the need of customers while minimizing the economic losses [11].

The target acquisition and detection algorithms are roughly as classification-dependent algorithms and regression-based algorithms. First, classification-dependent algorithms use two processing steps, first defining and selecting regions that are important to an image, and then organizing these regions into CNNs. Region-based RCNNs like Faster R-CNN [12] and Mask R-CNN fall into this category. Second, regression-based algorithms do not select regions of interest, but instead in just a single sweep of the algorithm estimate bounding boxes and clusters for the entire image. The YOLO family of algorithms, known for offering maximum speed and precision for the detection multiple objects in one image, and SSD, mainly used for real-time object tracking, fall into the regression-based family. Yolo models, SSD and Faster R-CNN offer the added benefit of increased speed and high accuracy in target detection. Yolov3 [13] uses a multi-scale prediction mechanism that helps improve the efficiency of detecting small targets (since it has three layers of detection for detecting large, medium and small targets), deepening the network for improved accuracy, making it more accurate and about 100 times faster than Faster RCNN.

To put it in better perspective, the authors in [20] surveyed the different approaches for industrial defect detection and classification. They reviewed many automated visual-based defect detection mechanisms, mostly applicable to material types like metals, ceramics, and textiles, about two (2) main classes of defects were identified, namely; visible, and palpable defects. Artificial neural networks (ANN) were used in [21] to detect the defects on surfaces of hot-rolled steel strips and they achieved an accuracy of 80%. In [22] the authors deployed a mechanism based on colour structure, edge histogram, and inclusive of homogeneous texture mechanism as a feature extractor and for classifier, KNN as classifier was utilized, for the detection of defects on the surface of hot-rolled steel strips with an accuracy of about 76% [22]. The Support Vector Machine (SVM) was applied by the authors in [23] and [24] as a classifier in their inspection mechanism, and SVM showed an enhanced performance compared to ANN using hot-rolled steel samples with an accuracy of 90%. [25] used a hybrid edge-detection mechanism built on heuristics involving human inspectors to identify surface defects, which they used to calculate the features. SVM was used as a classifier to classify the surface images as either flawed or flawless; The system they developed was applied to some surface texture databases and delivered about 94.19% correct classification [25]. Circa 2015, in [26] a mechanism that combines RPN and Faster R-CNN for detecting objects in order to generate almost free region proposals was proposed. More recently, the authors in [27], proposed a unique system dedicated to the detection of defects on the surface of steel using a deep auto-encoder network with NEU dataset of steel surface defects and built on ANN. Their model delivered an accuracy of 83.44% in cross-validation [27]. Again in [28], the authors proposed a S-CNN to detect defects on steel surfaces in one step. Their method showed a significant reduction in the amount of data required for the training of the proposed model, achieving good accuracy of

about 92.55% on 80% of the NEU dataset used in training [28]. While the NEU database of steel surface defects is still in use, [29] proposed steel surface defect detection based on ANN and Gradient Boosting Classifier. These mechanisms provided accuracies of 75.27% and 92.50%, respectively.

Therefore, part two of this study proposes an optimized algorithm for target detection based on YOLO v3, focusing on achieving near real-time error detection. To do this work, an NVIDIA Jetson Nano with a TX1 graphics GPU device running a 64-bit Ubuntu 18.04 LTS OS with about 62GB of disk space. The open-source CVAT tool developed by Intel Corporation was used for data annotation and formatting to meet the YOLO 1.1 dataset format system. The study was accomplished by annotating the image datasets obtained from the production floor of Hatko Electronics, Istanbul, a major steel terminal manufacturer, based on the three detect classes (scratches, bent pin, and spot regions) identified by their R&D unit, preparing the dataset in the YOLO dataset formatting system, and configuration files prepared using a text editor. After the training and testing process, the performance of the optimized YOLO v3 algorithm showed great improvement in prediction and classification accuracy, detection time, and stability.

2. PROBLEM STATEMENT ONE: SURFACE DEFECTS ON HOT-ROLLED STEEL STRIPS

This section of the study focuses on answering the question on the development of a transfer learning-based algorithm intended for the detection of common defects on the surface of hot-rolled steel strips using pre-trained VGG19 model using dataset from the NEU database of steel defects.

2.1. Motivation of Study

The increasing complexity of manufacturing systems and dynamic product designs, have placed a new burden on the quality control and inspection strategies of steel manufacturing entities, as the human-in-the-loop components of quality control and inspection have become inadequate to meet present day industrial inspection needs. Automation of these processes have therefore become more than inevitable, and a must now. Due to the lack or shortage of large datasets in the steel surface defects domain, from real production lines, and the requirement of real-time, fast converging, and production-ready models, an optimized implementation is being proposed in this study. With deep convolutional neural networks (DCNN), steel surface defects detection turnaround time will be greatly enhanced, increasing production output and customer satisfaction.

The desire to develop deep convolutional neural networks that are optimized to deliver high performance metrics, eliminate overfitting. The central motivation for this work is the use of limited training data while simultaneously applying data augmentation techniques and transfer learning primed with hyper-parameter tuning approaches for steel surface defect detection.

2.2. Contribution to Knowledge

The author envisages that this work will contribute immensely to the available knowledge-base in literature as highlighted in the section that follow;

1. Contribute to the available pool of knowledge on defect detection of steel surfaces systems using transfer learning in the steel industry by providing new insights, paradigms and architectures that are optimal and cost-effective. This is envisaged to be so, considering that the limitations of adequate training data, training and inference time, uniqueness of industrial environmental conditions, and randomness of steel surface defects, will have found solutions, as a result of this work.
2. The findings from this study will trigger more research interests within the research community and academia, to adapt the research output from this work to other industrial problem domains, investigate its applicability and suitability to such domains.

3. The study will also help the steel industry improve their product quality offering, satisfy customer needs, reduce material waste, gain on reduced operating cost, while steel products end-users will benefit from cheaper and high quality steel products.

4. The original knowledge gained through this research can be disseminated to the wider world through conferences, workshops, seminar presentations, and also the publication of papers in quality academic journals as pathways of research impact.

2.3. Classes of surface defects in hot-rolled steel

Dataset utilized in this study herein presented is derived from the publicly available benchmark dataset derived from NEU database [8]. There are about six (6) common categories of surface defects found on strips of hot rolled steel.

Figure 2.1. to Figure 2.6. shows examples of hot-rolled steel strips with cracks, inclusions, stains, pitted surface, rolled-in scale, and scratches surface defects, respectively.

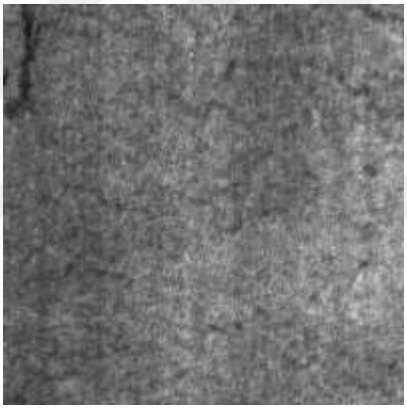


Figure 2.1. Example of crazing surface defect

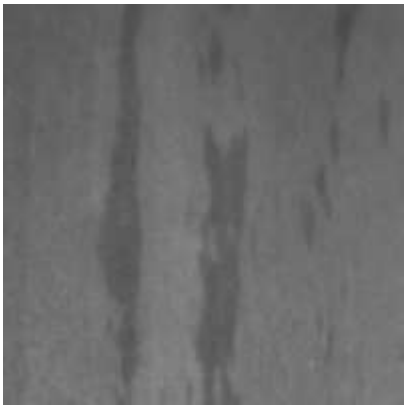


Figure 2.2. Example of inclusion defect

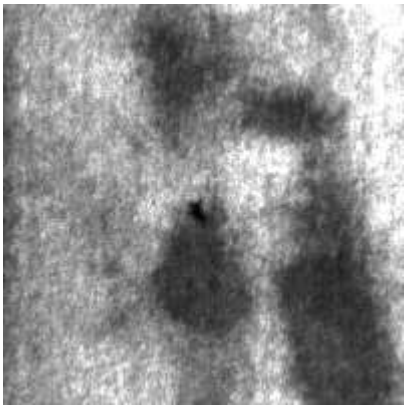


Figure 2.3. Example of patches defect

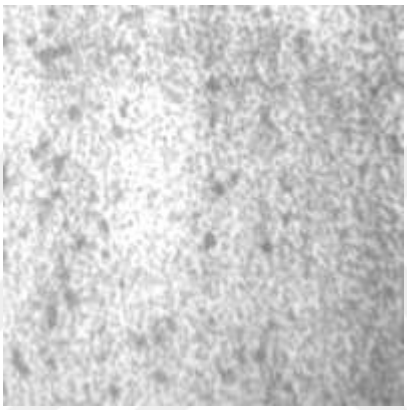


Figure 2.4. Example of pitted surface defect

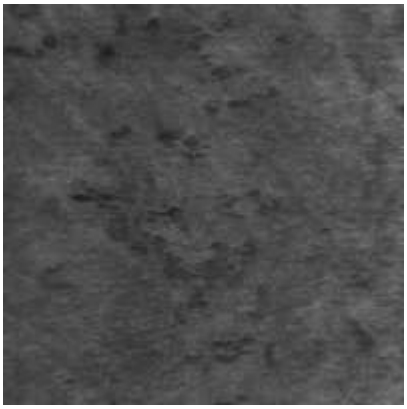


Figure 2.5. Example of rolled-in scale defect

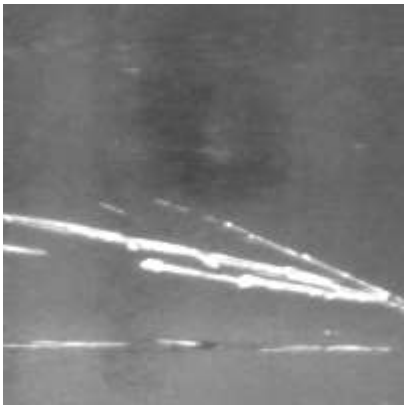


Figure 2.6. Example of scratches defect

3. MATERIALS AND METHODS

To realize this study, a dataset comprising of 1,800 greyscale images split into three (3) for training, validation, and test, respectively, as shown in Table 3.1. laid out in 200 x 200 pixels.

Table 3.1. Sample of study dataset composition

	Training	Validation	Test
Inclusion (In)	285	12	3
Patches (Pa)	285	12	3
Pitted surface (Ps)	285	12	3
Rolled-in scale (Rs)	285	12	3
Scratches (Sc)	285	12	3
Crazing (Cr)	285	12	3
Sub-Total	1710	72	18
Grand Total		1800	

3.1. Data Augmentation

Prediction accuracy of supervised deep learning models is tightly hinged on the amount and spread of data available for use in training. The relation between the deep learning models and amount of training data required is analogous to that of the relation between rocket engines (deep learning models) and considerable amount of fuel (huge amounts of data) required for the rocket to complete its mission (success of the deep learning model) [12].

Another approach to handle the problem of scarce data is the application of various transformations to the existing data to improvise new data points. This approach of improvising new data points from existing data is termed data augmentation. The data extension can be used to meet the above requirements; the variety of training data and the amount of data. Besides these two, extended data can also be used to address the class imbalance problem in classification tasks. It is known that deep conventions (DCNNs) possess a property called translation invariance. Translation invariance implies that even if we translate the model inputs, the CNN model will still be able to recognize the class to which the input image belongs. Translation invariance comes from the

pooling operation. The pooling operation replaces the convnet's output at a specific point with summary statistics of nearby outputs, such as maximum in the case of Max Pooling. Max pooling replaces the output with the maximum, so the values of most of the pooled outputs are not affected even if we slightly adjust the input.

Some of the data augmentation transformations implemented in this work are;

i. Width_shift_range
ii. Height_shift_range
iii. Shear_range
iv. Zoom_range
v. Channel_shift_range
vi. Horizontal_flip

Also, the image resolutions were retained as 200x200 to align with the expected input of the convolutional base (conv_base) of the model.

3.2. Transfer Learning

The model in this study is based on the widely acclaimed game changer theory and concept in deep learning and computer vision known as transfer learning. Transfer learning is a typical convolutional neural network implementation premised on the basic assumption that networks trained on very large (mostly in the millions) natural image datasets do learn enough and adequate generalization features (both high and low-level features) and what they have learned can be exploited and deployed in more domain-specific implementations in this study, detection of steel surface defects [12, 13]. Figure 3.1. Shows the structure of a pre-trained model.

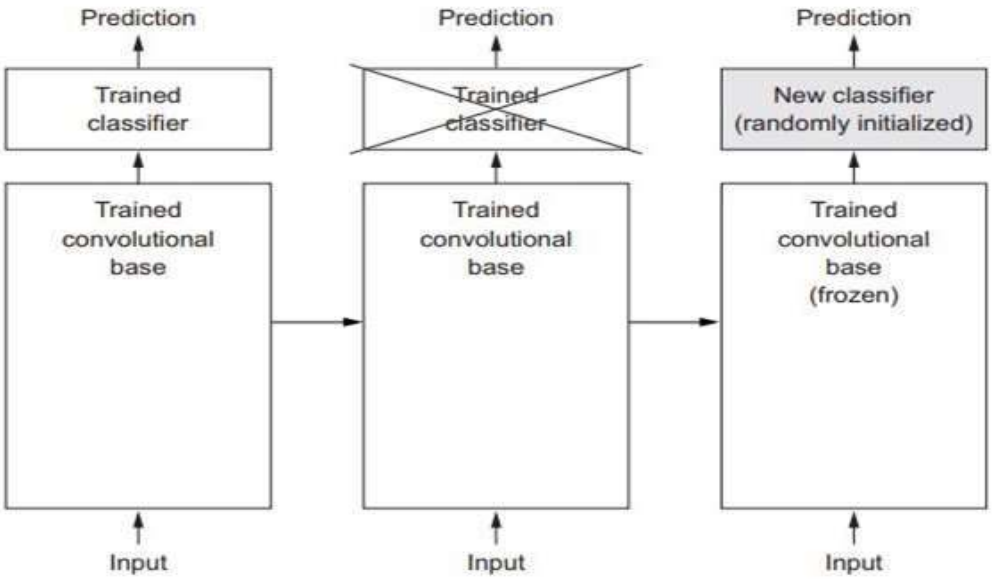


Figure 3.1. A pre-trained model showing the convolutional and dense layers

3.2.1. Optimized Model Architecture

The network architecture of the pre-trained VGG19 network is shown in Figure 3.2.

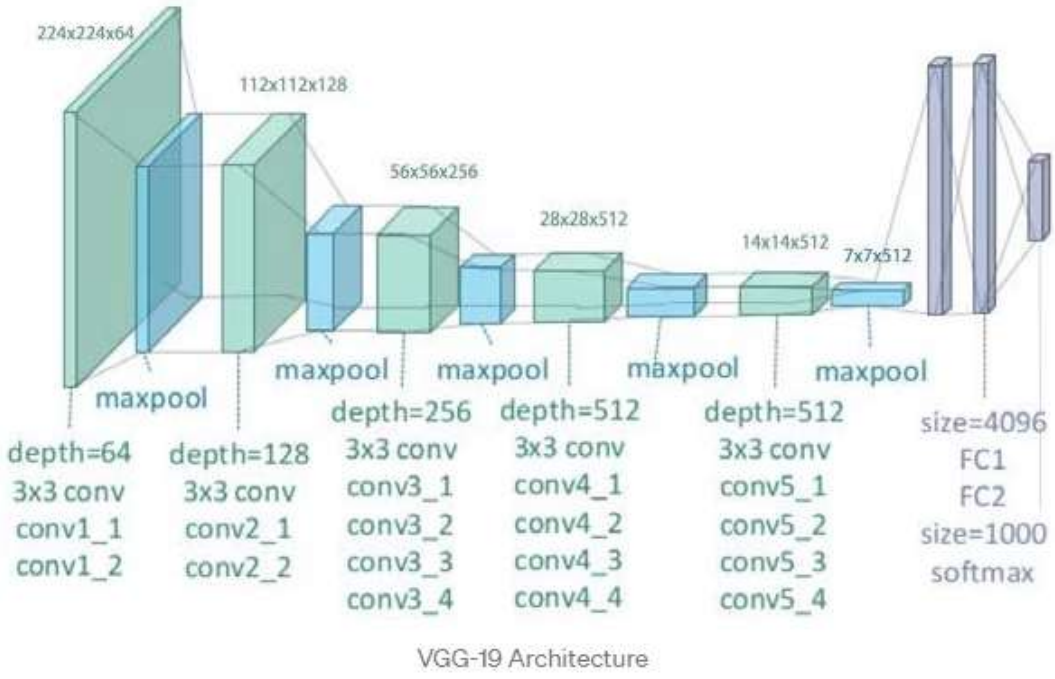


Figure 3.2. Architecture of VGG-19 pre-trained model

The optimized model proposed in this study herein comprises a VGG-19 with a convolutional base (conv_base) with four tightly connected (FC) layers, and the fully connected layers are reconstructed as shown in Table 3.2.

Table 3.2. Architecture of proposed VGG-19 model

Input (200 x 200 grayscale image) New
VGG-19 conv_base
FC-256 (New)
FC-256 (New)
FC-256 (New)
FC-256 (New)
Soft-max

During the model training process, training images are input into the optimized model and scaled or validated to a resolution of 200200 gray images. Next, the scaled images are passed through a VGG-19 conv_base that includes 20,024,384 parameters previously stored after training with the ImageNet database of millions of natural images. Finally, four fully connected (FC) layers are built on the convection base (conv_base). In this study, the convolutional layers of the VGG-19 are frozen (its trainable is set to non-trainable so they will not get retrained in the process), while the fully connected (FC) layers are replaced by four (4) new layers in line with transfer learning concept. The parameters of convolution layers are large (about 20,024,384) with the prior weights already trained on the ImageNet dataset. The convolution layers have a strong advantage of extracting features from edges of images and even contours [14]. The proposed optimized network architecture is able to clearly and robustly extract steel surface defect features considering the depth of the neural network, which is judged to be sufficient. In addition, the optimized model configuration is less likely to over-fit because the hyper-parameters are methodically chosen and tuned. The first fully connected layer has about 256 channels (neurons) followed by a dropout layer for reducing the chances of overfitting. ReLU is used in all of the four (4) fully connected (FC) layers. In particular, the bottom FC in the herein optimized model mainly represents an N-class predictor, where N represents the number of class labels in training dataset. The FC layer as presented in this study includes six channels to account for six classifications of steel surface defects within the dataset (as included in the NEU dataset). The FC layer (last layer) generates the final classification results, that is the softmax layer (activation layer). It should be emphasized that the logistic function which is designed to map a p-length vector comprising real values to a k-length vector with values, is generalized by the Softmax function. The combination of the categorical

cross-entropy loss function and softmax used in this study is unarguably one of the most commonly used components for monitoring in DCNNs [15].

3.2.2. Model Activation Function

Model activation functions are generally classified into two (2), namely;

i. The linear or identity activation function and,

ii. Non-linear activation functions such as the Sigmoid, Tanh, ReLU as Figure 3.3 shows, etc.

Nonlinear activation functions are primarily used in neural network designs and implementations. This approach helps neural networks learn from complex data, compute and learn almost any function that represents a question, and offer accurate predictions [16, 17]. This allows backward propagation since nonlinear activation functions have a derivative function related to the inputs. The ReLU activation function was used in the FC layers to eliminate the vanishing gradient problem and achieve optimal computational performance [16 - 19]. In the last layer, the softmax activation function, as shown in Figure 3.4. shown, used to calculate the event's probability distribution over n distinct events. Ability to handle multiple classes is the main benefit of using the softmax activation function.

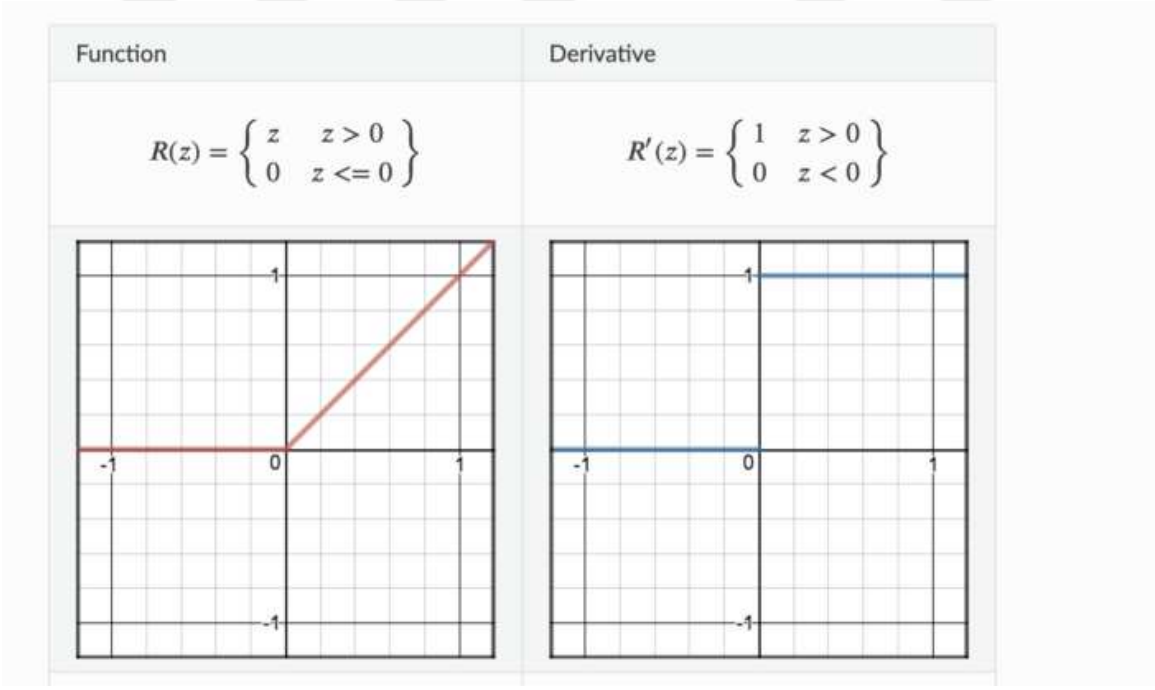


Figure 3.3. ReLU activation function curve

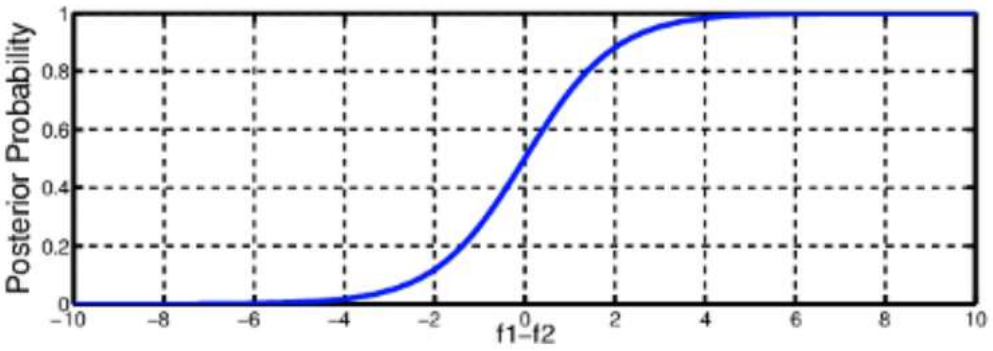


Figure 3.4. Softmax activation function

3.3. Optimized VGG-19 Model Training

In this study, the RMSProp algorithm is used for optimization with a batch size of 32. The RMSProp is an adaptive learning rate method proposed by Geoff Hinton in 2012 [6]. The RMSProp has proven to be very effective and is therefore very popular among researchers and developers within the deep learning ecosystem. In this study, the learning rate is set at $1 \times e{-}4$. Also, dropout rate of 0.2 is used to regulate the first three fully connected layers, reducing overmatching.

Referring to Figure 3.5. the convolution base is frozen to ensure that the parameters of the convolutional layers are not updated during training. With this configuration, only four fully connected (FC) layers are trained to predict steel surface defects. Greater depth and smaller size convolution filters are some of the advantages of the VGG-19 model, resulting in the optimized model converging after just a few epochs (15 in this study). As a result, the time it takes to train the model is reduced, and the model can also be run on a standard industrial computer with less complexity and setup effort without sacrificing performance. In the training phase, the preprocessed and transformed images, sized as 200 x 200 images, from the data augmentation pipeline are fed to the model.

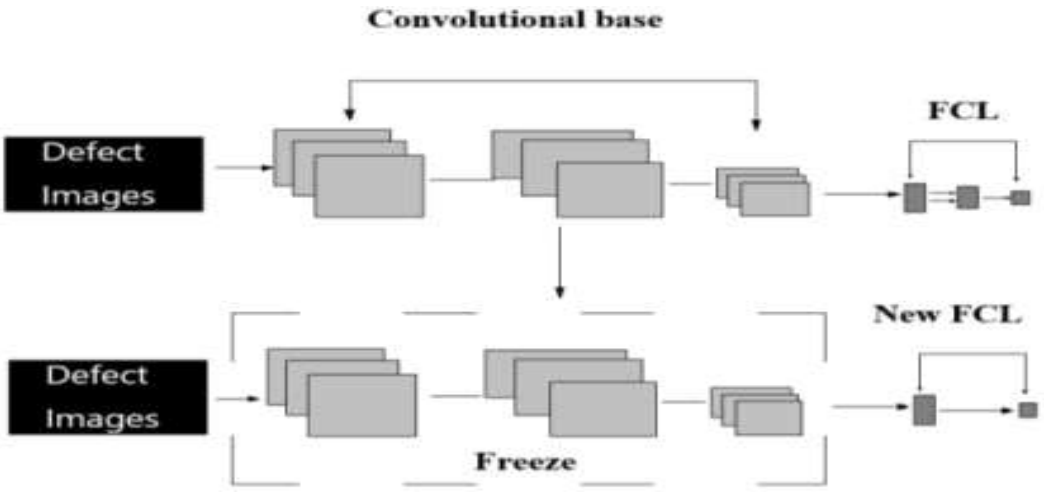


Figure 3.5. Modified VGG-19 model

4. RESULTS AND DISCUSSION

During training, 1710 images of steel surface defect comprising of six defect classes was used for training, while 72 images was used for validation. Figures 4.1. and Figure 4.2. show the accuracy and loss curves, respectively, for both training and validation, after 15 epochs. This result indicates that the optimized model is capable of excellent performance.

From the results, training and validation accuracies of the optimized model are 91.31% and 97.22%, respectively. Testing accuracy stood at 93.1%.

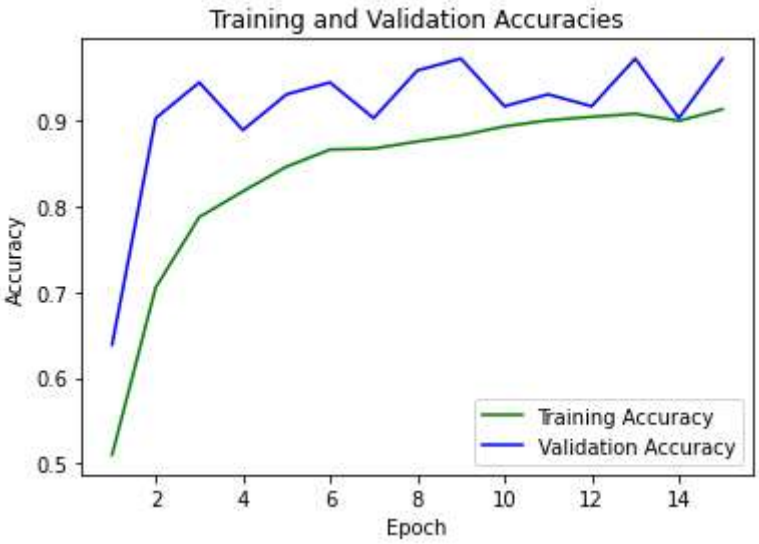


Figure 4.1. Experimental training and validation accuracy plot

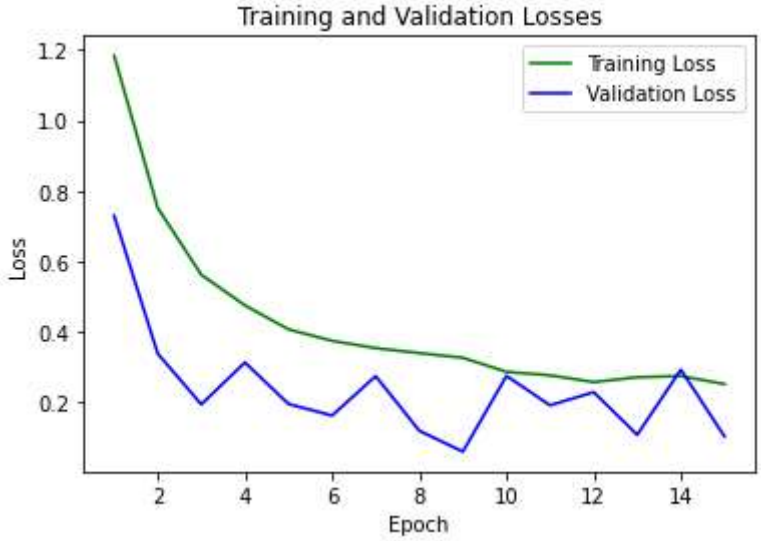


Figure 4.2. Experimental training and validation loss plot

Figure 4.3. shows the Jupyter Notebook screenshot of the training and validation steps and the accompanying results of performance metrics.

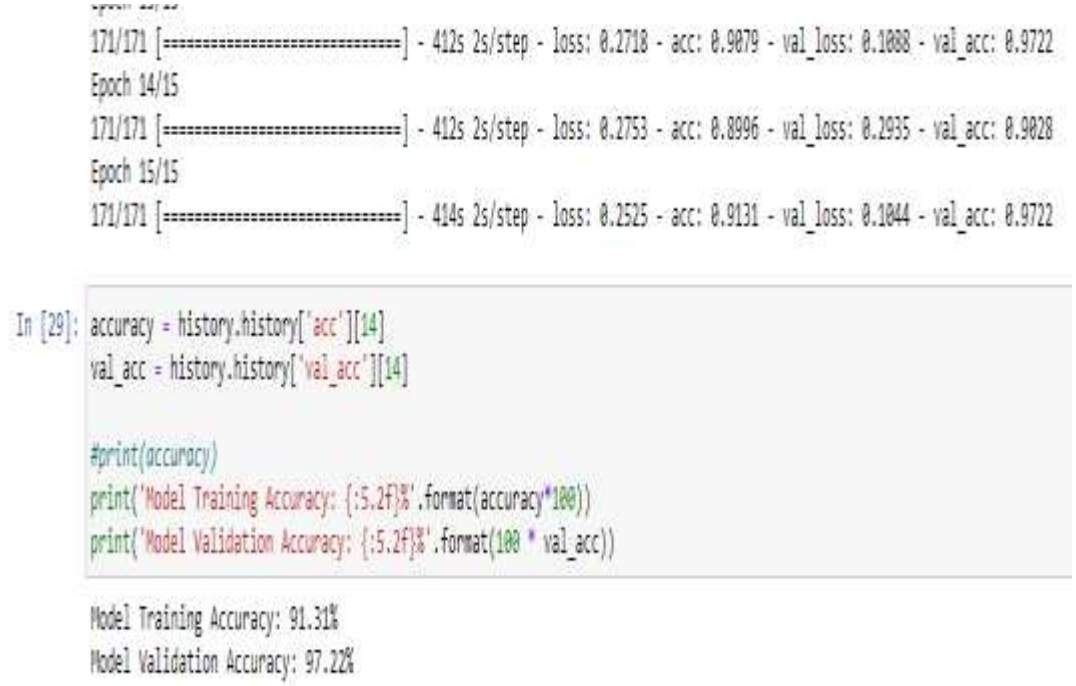


Figure 4.3. Screenshot of Jupyter notebook showing training and validation steps and results.

4.1. Optimized VGG-19 Model Test

The optimized model was fed 18 images that it had never seen before. This included 3 images per error class. The output form of the optimized model is shown in Figure 4.5 and Figure 4.4. shows the prediction output of the optimized VGG-19 proposed in this study.

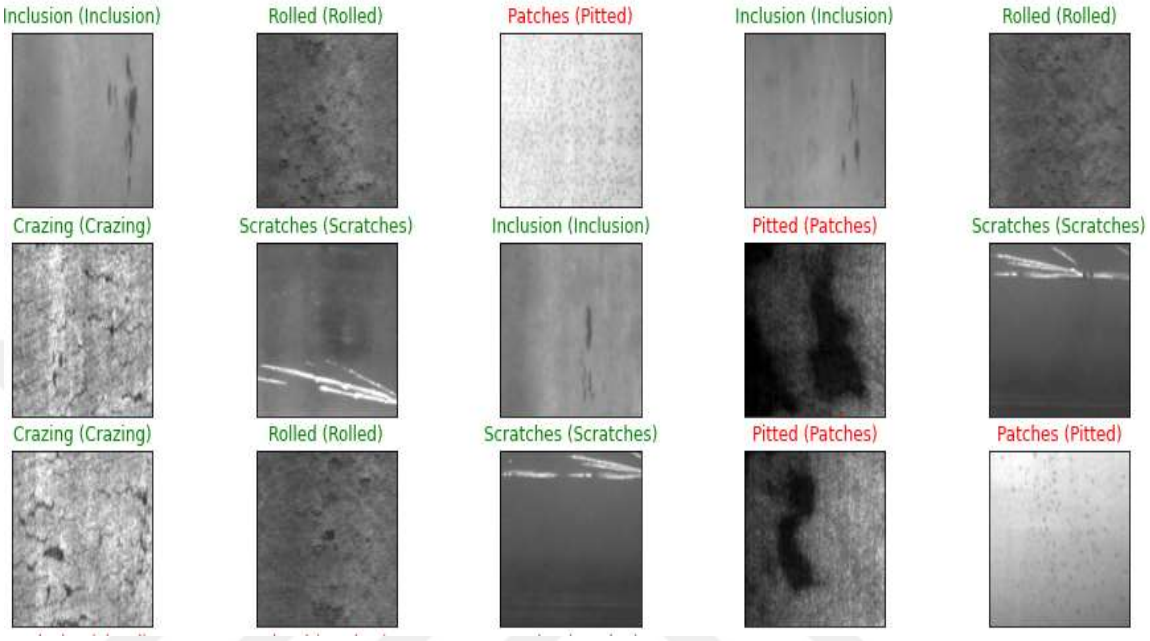


Figure 4.4. Prediction output of Proposed VGG-19 model

```
In [12]: no_of_classes = len(np.unique(y_test))
         no_of_classes

Out[12]: 6

In [13]: from keras.utils import np_utils
         y_test = np_utils.to_categorical(y_test,no_of_classes)

In [14]: # We just have the file names in the x set. Let's load the images and convert them into array.
         from keras.preprocessing.image import array_to_img, img_to_array, load_img

         def convert_image_to_array(files):
             images_as_array=[]
             for file in files:
                 # Convert to Numpy Array
                 images_as_array.append(img_to_array(load_img(file)))
             return images_as_array

         x_test = np.array(convert_image_to_array(x_test))
         print('Test set shape : ',x_test.shape)

         Test set shape :  (18, 200, 200, 3)
```

Figure 4.5. Test data shape screenshot.

4.2. Discussion and Future Works

Transfer learning is a widely used and recommendable deep learning methodology in the computing research community. This is mostly attributable to its ease of implementation, low compute cost, faster model training.

The authors in [7], [8] and [9] recommended the search for an implementation method based on transfer learning for detecting surface defects on hot-rolled steel strips. Their S-CNN model achieved 92.55% accuracy on 80% of the NEU dataset used in training, using high-channel, feature-rich sensor data. From experimental results, the optimized VGG-19 model proposed herein in this study shows higher performance compared the S-CNN model, and Xception model with 94% accuracy, and is faster too. This result shows that the model proposed herein offer enhanced performance in comparison with other models in the literature per defects on the surface of hot-rolled steel strips and can be used as a high-performance algorithm in the steel industry.

The next research direction will be the enhancement of the algorithm proposed in this study in terms of accuracy, speed, and then most importantly real-time operation of the detection algorithm with an option for remote deployment and use. The author looks in this direction as this has the potential to offer added benefits to hot-rolled steel strips manufacturers.

5. PROBLEM STATEMENT TWO: DETECTING DEFECTS ON STEEL TERMINALS USING AN OPTIMIZEDYOLOV3

Problem statement two of this work is focused on developing a high-performance defect detection algorithm using YOLO v3 for use in detecting surface defects on steel terminals.

5.1. Steel Terminals

Steel terminals are mostly wire-end termination end-points, and are a major component that is highly sought after and utilized in the automotive, aeronautic, shipbuilding, electrical/electronic, and white goods industrial domains. In the application areas where copper coated terminals and connectors are used, steel terminals are mostly the base components used in these use cases. As can be seen in Figure 5.1., a copper connector made out of a steel terminal.

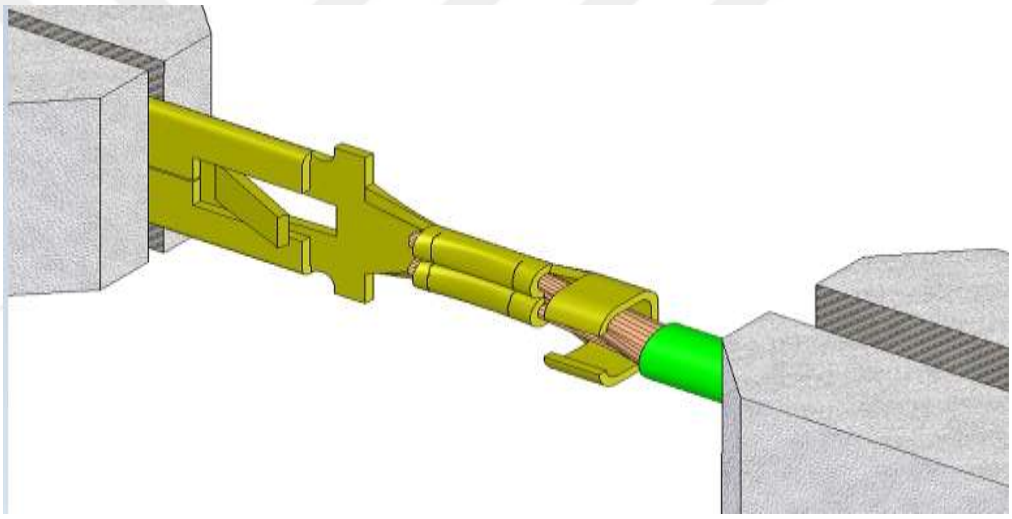


Figure 5.1. Example of a finished copper-coated steel terminal

5.2. Steel Terminal Defect Classes and Causes

From the data acquired from a steel terminal production line and used in this thesis, a total three (3) classes of defects were identified. These defects classes are;

5.2.1. Bent Pin Defect

As shown in Figure 5.2., the pins on steel terminals can become bent when it comes in contact with excessive force or obstruction on its path in the production line, and therefore unfit for its intended use.

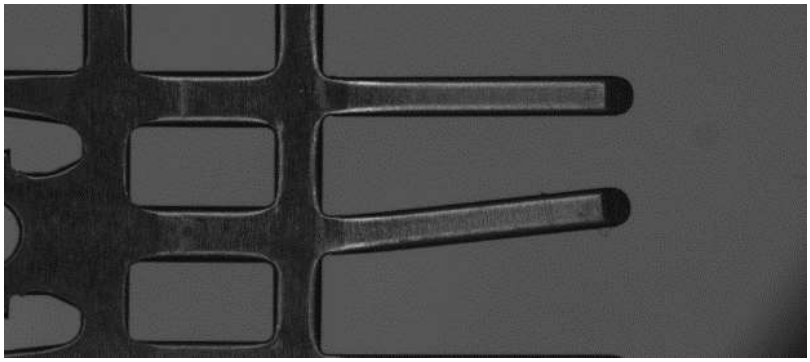


Figure 5.2. Example of steel terminal with a bent pin defect

5.2.2. Scratches Defect

These kinds of defects occur on steel terminals during the cutting and shaping stage where a misalignment in the cutting tool or momentary vibration makes unintended scratches on the surfaces of the steel terminal as it runs through the line, and is as shown in Figure 5.3.

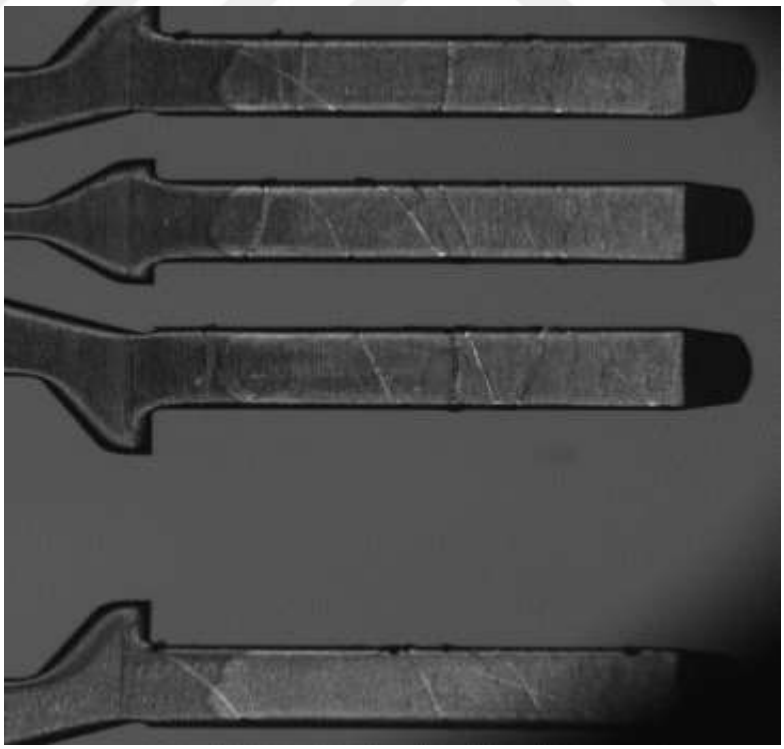


Figure 5.3. Example of steel terminal with scratches defect.

5.2.3. Spot Region Defect

During surface treatment of steel terminals, unwanted drops of the treatment substance sometimes get splashed on the steel terminal surface or the intended surface treatment substance is applied beyond the recommended proportion. This leads to a defect that renders the defective terminal unsuitable for use and has to be isolated from the rest of the batch in the production cycle, as shown in Figure 5.4.

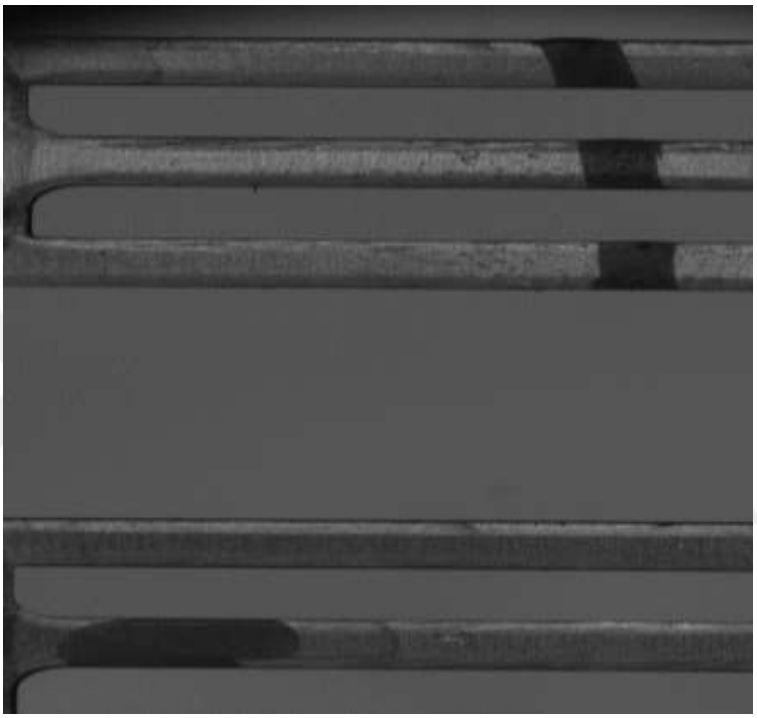


Figure 5.4. Example of steel terminal with spot regions defect

6. YOLO V3

The outstanding feature of Yolo v3 [30] is the use of a single convolution network to predict target bounding boxes and their corresponding probabilities. The backbone network for Yolov3 is Darknet-53 [17], with a unique variant called darknet-53.conv.74 that is well suited for training object detection networks, which is used alongside the COCO datasets used in training the original Yolov3 network are used, use custom records.

The probabilities of these bounding boxes are used as weights and the network performs its detection based on the final weights. This ensures that directly, the end-to-end output of the proposed model can be maximized, resulting in fast creation and processing of images [31]. About four descriptors can be used to represent the bounding box, namely:

1. Center point of a bounding box (bx, by)
2. width (bw)
3. Height (bh)
4. The value c refers to an object class

The overall Darknet-53 architecture is as shown in Figure 6.1.

	Type	Filters	Size	Output
	Convolutional	32	3 × 3	256 × 256
	Convolutional	64	3 × 3 / 2	128 × 128
1×	Convolutional	32	1 × 1	
	Convolutional	64	3 × 3	
	Residual			128 × 128
	Convolutional	128	3 × 3 / 2	64 × 64
2×	Convolutional	64	1 × 1	
	Convolutional	128	3 × 3	
	Residual			64 × 64
	Convolutional	256	3 × 3 / 2	32 × 32
8×	Convolutional	128	1 × 1	
	Convolutional	256	3 × 3	
	Residual			32 × 32
	Convolutional	512	3 × 3 / 2	16 × 16
8×	Convolutional	256	1 × 1	
	Convolutional	512	3 × 3	
	Residual			16 × 16
	Convolutional	1024	3 × 3 / 2	8 × 8
4×	Convolutional	512	1 × 1	
	Convolutional	1024	3 × 3	
	Residual			8 × 8
	Avgpool		Global	
	Connected		1000	
	Softmax			

Figure 6.1. Overall architecture of Darknet-53

6.1. YOLO V3 Network Structure

Input images are divided into grids by Yolo v3, e.g., (3 x 3) grids. On each grid, object classification and localization operations are hereby applied, then for each object Yolov3 predicts the bounding boxes and class probabilities, correspondingly. To train Yolo v3 with custom datasets, labeled data with its annotations are fed to the model. Data annotations for each image, and the image files intended for use in training are contained in a obj_train_data folder, text files (train.txt, valid.txt, and test.txt) contain the directory path for each image used for training, validation and test, obj.data file contain the paths for the training, validation, test, and obj.names files, while the class labels are housed in the obj.names file. Configurations for training and testing are contained in the yolov3-custom.cfg file.

For example, suppose the image is divided into a (3 x 3) grid and there are a total of 6 classes, to which each object can belong, then the y-label for an 11-dimensional vector for each of the class grid cells [32] as Table 6.1 shown.

Table 6.1. Example of a Grid cell with 11-dimensional vector

y =	pc
	bx
	by
	bh
	bw
	C1
	C2
	C3
	C4
	C5
	C6

i. pc indicates the probability that an object is present in the grid.

ii. The bounding box coordinates assuming an entity exists in the grid are represented by bx, by, bh, and bw.

iii. The classes are represented by C1, C2…C6.

Assuming there is an entity in the grid, the pc value in the 11-dimensional will be 1 and if the entity belongs to say class C3, its value will be C3 = 1 while C1, C2, C4, C5, and C6 = 0. This is

represented in Table 6.2.

Table 6.2. Example Y label of predicted classes.

y =	1
	bx
	by
	bh
	bw
	0
	0
	1
	0
	0
	0

The dimension of the rendered grid is (3 x 3 x 11). The probability of having different objects in similar cells can be greatly reduced by basically increasing the number of grids.

Structurally, YOLO v3 consists of 106 layers composed of three (3) different layer types [31]. First, the residual layer (res_unit) is hence formed when the activation can be easily propagated to layer that is lower in the neural network, and is known to eliminate gradient-vanishing in the Darknet-53 network. The Layer 1 outputs are added to the Layer 2 outputs, which are the basic building blocks of YOLO v3 and this contain the convolution, the batch normalization (BN) and the leaky RELU functions. Second, the detection layer performs 3-scale detection, and the grid size is increased for detection. Third, the spatial resolution of an image is increased via an up-sampling layer just before the image is scaled, followed by a concatenation operation that concatenates the outputs of the current layer with the previous layer. Figure 6.2 shows the basic architecture of YOLOv3 showing the 3 different unique layer shapes [32].

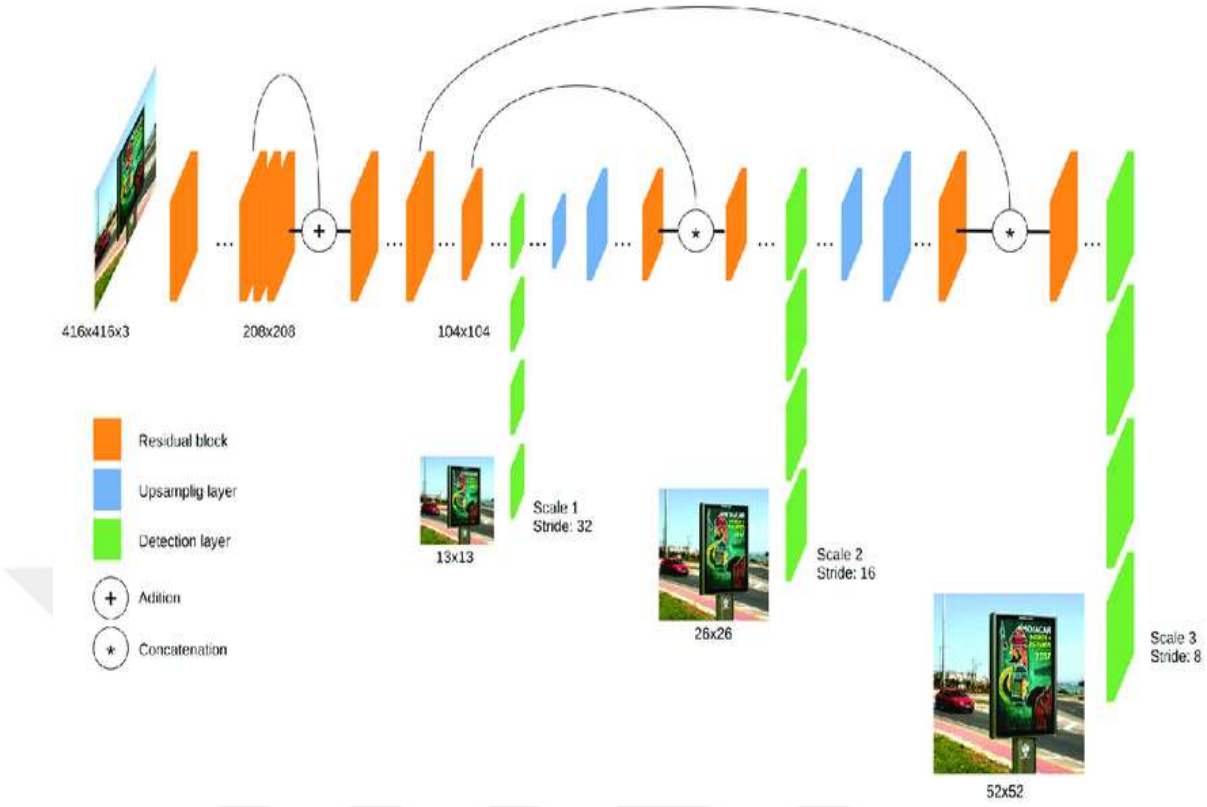


Figure 6.2. Basic architecture of YOLO v3

6.2. YOLO V3 Target Detection

A multi-scale detection process is employed by YOLO v3 for target detection, as shown in Figure 3.3., and prediction. It uses 3 different scales of prediction. Three different sizes of feature maps, with strides of 32, 16, and 8 respectively, are detected by the detection layer. This implies that with an input size of 416 x 416 x 3 (YOLO v3 default input size), the detections are on scales of 13 x 13, 26 x 26 and 52 x 52. The goal is the large-area target recognition to acquire a feature map of 1/32 (13 x 13) after many convolution processes immediately succeeding the 79th layer with a clearly large receptive field. In this phase, up-sampling is applied to the result of the previous phase and then concatenated (tensor stitching) with the result of the 61st layer. A 1/16 (26 x 26) feature map appropriate for medium scale target detection is derived by convolution. To detect small-scale targets, the results from the above step are passed through an up-sampling phase, concatenated with results from the 36th layer and then convolved to obtain a feature map of 1/8 (52 x 52) that has a really small field of view. For the COCO (original Yolo v3) category, each of the boxes must recognize 80 classes, making the depth of the feature map 255 (i.e. at is; ((80 +5)3)). Formula (6.1) can be used to the depth of the feature map, and Figure 6.3 is shows the multi-scale prediction mechanism of Yolov3.

$$\text{feature map depth} = (\text{Number of classes} + 5) \times 3) \quad (6.1)$$

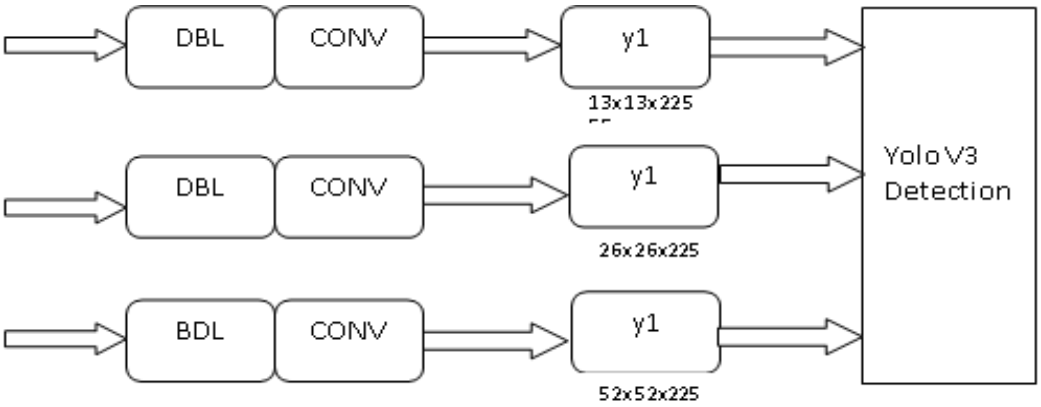


Figure 6.3. YOLO v3 multi-scale prediction [22]

6.2.1. Making Predictions

YOLO v3 make use of the K-Means (that is mean clustering) for the determination of the priori anchor size in the image. Considering that the output of the YOLO v3 is a feature map consisting of three-scales, with three previous frames corresponding to small, medium, and large sizes, resulting in about 9 sizes of priori boxes. The nine clusters for the COCO dataset with their respective prior anchor is shown in Table 6.3, with different scales of feature maps corresponding to each of the different size prior anchors.

Table 6.3. Example COCO dataset 9 kinds of prior boxes

	Feature maps of different sizes		
	13*13	**26*26**	**52*52**
Receptive grid	**Big**	**Medium**	**Small**
Priors anchor	(116x90)	(30x61)	(10x13)
	(156x198)	(62x45)	(16x30)
	(373x326)	(59x119)	(33x23)

Preceding the prediction of the bounding boxes, YOLO v3 performs an objective assessment of the area around the previous anchor point using logistic regression [34] to evaluate if the area is the intended target area before predicting the bounding boxes. The model will not predict a target area if it is not obviously the best, it will only predict the target area or region which has the highest probability of the target being present. This allows the model to eliminate unnecessary previous anchors, reduce computation time and therefore run faster.

To explain more clearly how the process of converting the network's output is converted to obtain bounding box predictions, Equation (6.2) to Equation (6.5) come in handy. The width and height of the prediction representing the network output are bx, by, bw, bh, while the center coordinates x, y, cx, and cy represent the top left coordinate of the grid, and the dimensions of the anchor boxes are pw and ph. [18].

$$bx = \sigma(tx) + cx \tag{6.2}$$

$$by = \sigma(ty) + cy \tag{6.3}$$

$$bw = pw \times e^{tw} \tag{6.4}$$

$$bh = ph \times e^{th} \tag{6.5}$$

The x and y coordinates of the center of the grid object and the grid coordinates are of the form bx and by, while bh connotes the ratio of height of the bounding box to the height of the corresponding grid cell, while bw represents the ratio of the width of the grid cell. To achieve normalization, YOLO v3 uses the sigmoid function rather than the softmax regression method. This is owing to the fact that the softmax method mostly increases the probability of the most probable (possible) class while suppressing the value of every other category, a situation that is not idea for multiclass classification tasks.

This implies that the sigmoid function is more accurate than other normalization methods, such as the softmax regression method, and is therefore widely used.

6.2.2. Intersection Over Union

The concept of intersection over Union (IOU) is aimed at measuring the intersection of the expected bounding box and the ground truth bounding box over their union. For example, consider the expected bounding boxes, say of a car, and the ground truth, Figure 6.4.

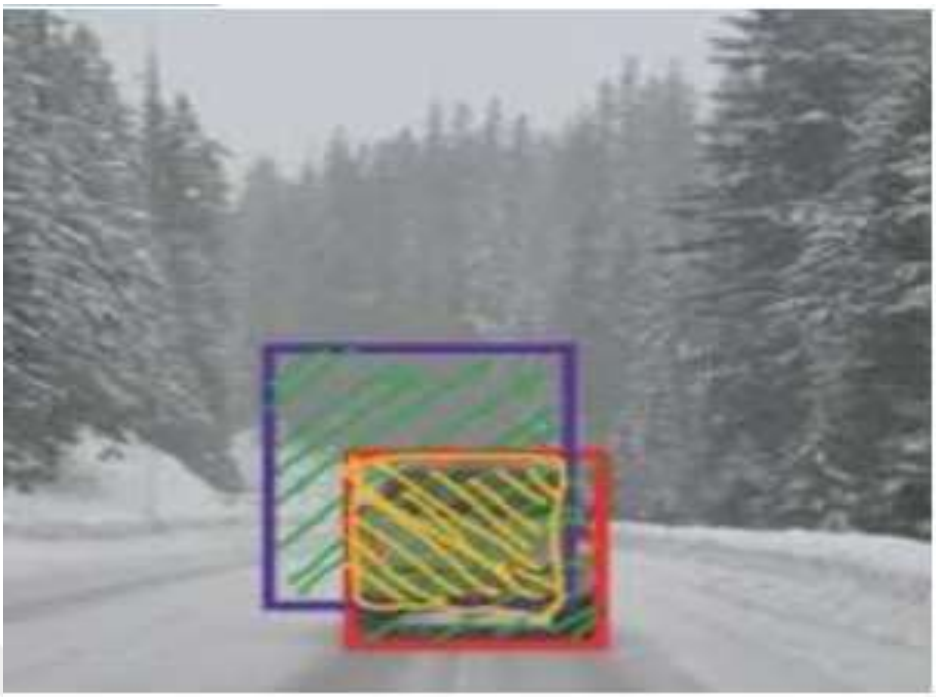


Figure 6.4. Intersection over Union [32]

In Figure 6.4. the red box represents the ground truth bounding box and the expected bounding box is represented by the purple box. The area of intersection over the union of these two boxes is determined by IOU. [32] i.e. the yellow shaded area in Figure 6.4. In general, formula (6.6) shows how IOU are calculated.

$$\text{IOU} = \text{Area of intersection} \div \text{Area of the green box} \tag{6.6}$$

If IOU > 0.5, then the estimate is reasonable. It has to be noted that 0.5 is an arbitrary limit and can be changed if explicit problems suggest or require it. Raising this limit higher will deliver better predictions and the model's prediction confidence will be high too [19].

Figure 6.5. shows a flowchart of the operation of YOLO v3 network the point of entry of input images, through target detection using bounding boxes, and the final predicted image output with its class label.

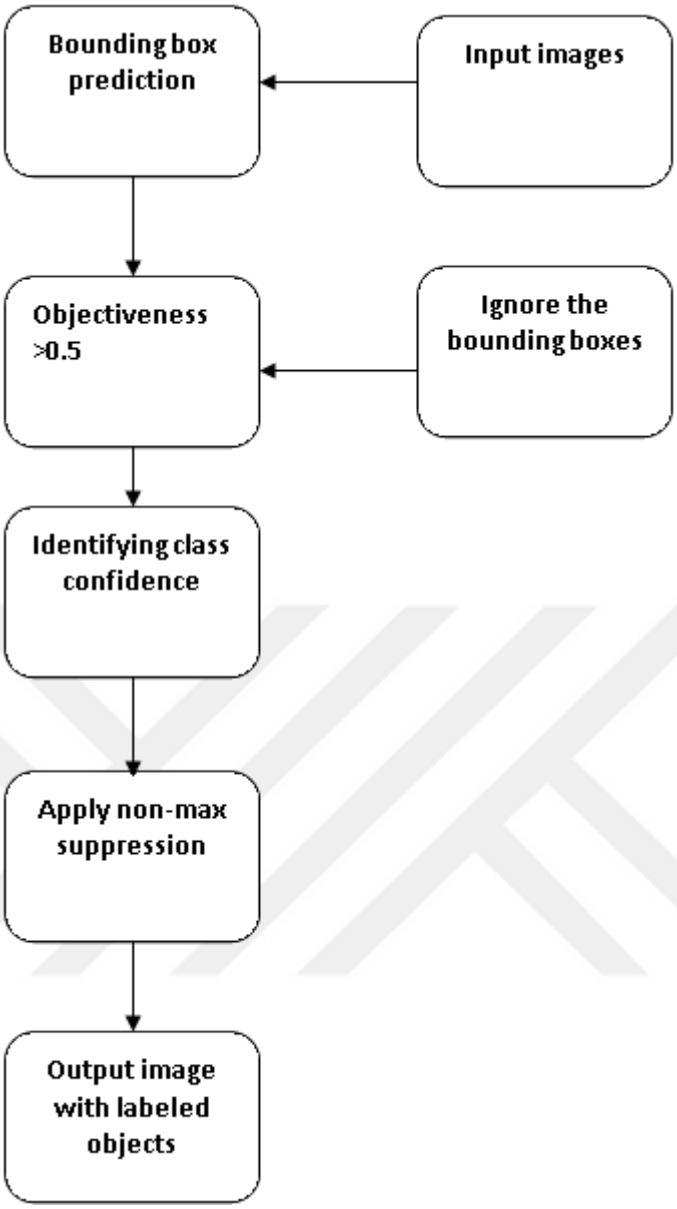


Figure 6.5. Flowchart showing YOLO v3 internal operation

7. MATERIAL AND METHOD

This section will highlight and detail some the materials and methods used in the actualization of this study, from data acquisition to model design and training.

7.1. Compute Resources

This study made use of some specific compute resources, both hardware and software. The hardware resources used are;

i. Jetson Nano Tegra X1 developer kit with an NVGPU, with power supply pack. The Jetson Nano is the main hardware platform used in the study as it provided the hardware and software resources plus operating system (Ubuntu) used in this study.
ii. Computer monitor
iii. HDMI cable
iv. USB keyboard
v. USB mouse
vi. IDS USB3 industrial camera (UI – 3270CP-C-HQ Rev.2) with a 1/18 Sony IMX265 global shutter, full HD with up to 80fps, and 3.2MP (2086 x 1542 px) resolution.

For software, Linux flavor operating system, Ubuntu was utilized, python programming language and the libraries needed for deep learning and computer vision modelling was used. Figure. 7.1. shows the hardware devices and their setup while Table. 7.1. shows the hardware specifications of Jetson Nano used during this study.

Table 7.1. Jetson Nano hardware specification.

Device Component	Jetson Nano Tegra X1 (NVGPU)/integrated
Memory	3.9GiB
Operating system	Ubuntu 18.04 LTS
Processor	ARMv8 Processor rev 1 (v8l) x 4
OS Type	64-bit
Disk Size	62.7GB
Deep learning framework	Darknet53 (Yolo v3) with CUDNN, OpenCV

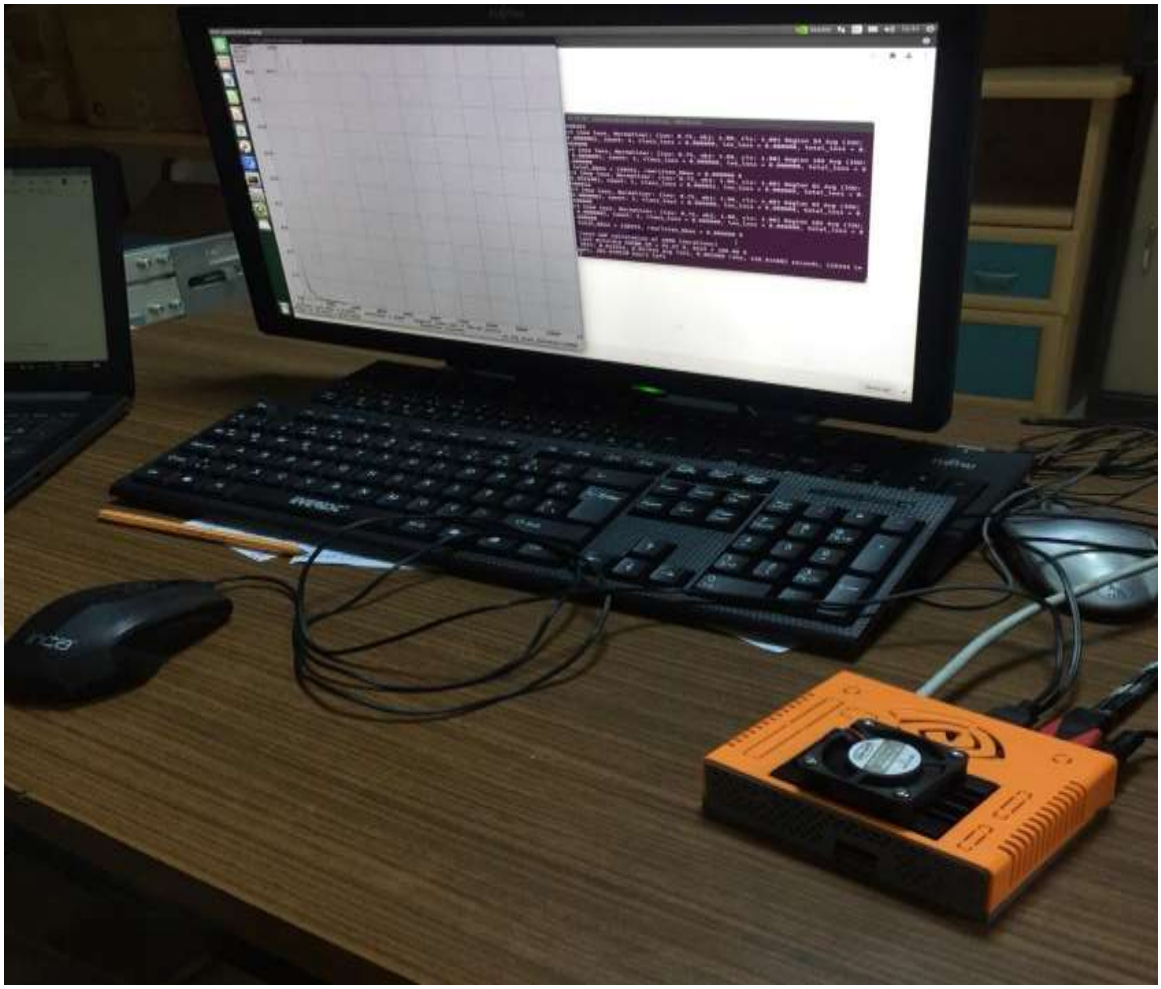


Figure 7.1. Jetson Nano and other hardware devises setup

7.2. Data Acquisition

Deep learning models have a very high appetite for large datasets and acquiring the right data in good enough quantity is always a major part and critical component of a computer vision project. This is so considering that the quality of a computer vision model depends entirely on the data it is trained with. There loads of free and paid datasets available for researchers to work with but industrial data from real production lines are almost impossible to get, as most industries do not release their data to the public to wade off competition and preserve their intellectual property.

However, this study had the privilege of the kind gesture of Hatko Electronics, Istanbul, that granted access to their production line for real-time data acquisition. The data used in this study was acquired raw from Hatko's production plant where they manufacture various designs and specifications of steel terminals for various companies across the globe. The data acquisition phase of this study took the author to Istanbul where he worked closely with the R&D and production team at Hatko for a period of four (4) days. Throughout the duration of data acquisition, the author

observed the production system at Hatko and the various product inspection points where steel terminals were automatically inspected for defects as it travelled along the production line. Figure 7.2. shows a steel terminal entering the inspection kiosk at Hatko Electronics.

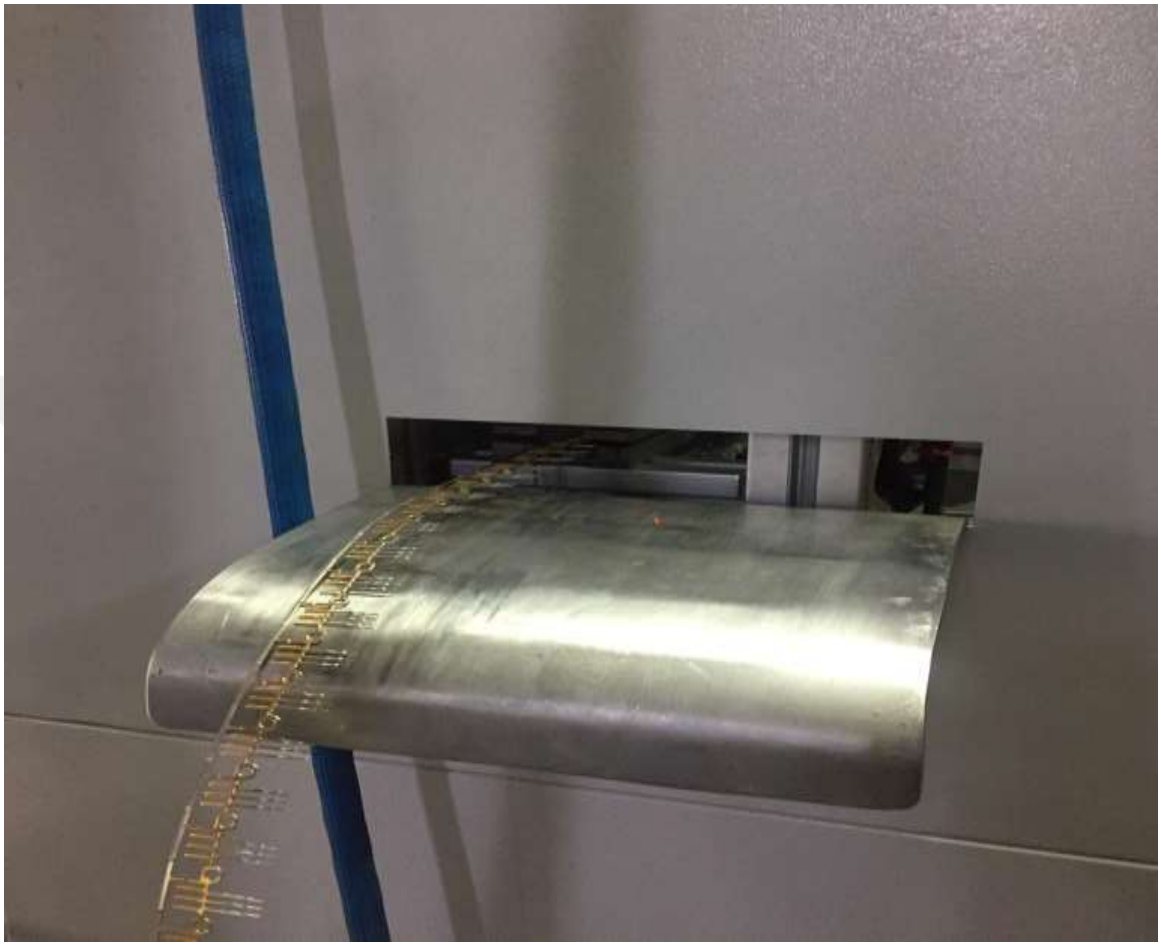


Figure 7.2. Example of steel terminal entering the quality inspection kiosk.

Inside the inspection kiosk, camera systems, lighting, and control systems are housed and operated to obtain live images of steel terminals as they journey through the production line, with a computer monitor displaying the images and their quality status. From the monitor screen the camera system can be operated and configured, while acquired images can be saved too, since the system is not configured to save images by default. For copyright issues, Hatko did not grant permission to take pictures of internal setup of the inspection kiosk and the camera system. For this study, an external hard drive storage device was connected to the control system and acquired images saved to this drive.

7.2.1. Data Preprocessing and Preparation

The file system in use by Hatko's quality control system have a unique way of saving images by putting each image inside a separate folder and with a unique name. this meant that if 500 image samples were obtained there will be 500 folders, each containing 1 image, and all the images had same file names.

For this reason, the images cannot be accessed straightaway, and a python script was created by the author to first extract all the images into a single folder and assign unique filenames to the images.

Two, all the images were manually inspected to be evaluate their quality in terms of resolution and defect class while then saving each image into separate folders according to their defect category – bent pin, scratches, and spot regions.

Three, for data preparation, the preprocessed dataset was then prepared to conform to the formatting standard required for this study. YOLO v3 uses a unique dataset format system consisting of;

i. Annotation file containing all the images and their individual annotations. Image annotations contained definitions of each image size, location, and the bounding box descriptions.
ii. Training, validation, and test files containing directory paths to all image samples to be used for training, validation and test, saved as train.txt, valid.txt, and test.txt files.
iii. An obj.names file containing the labels for the dataset defect classes, and
iv. An obj.data file containing the configurations and definitions for the train, valid, and test files, including the defect labels.

There are many different kinds of tools for image annotation such as Robo-Flow, labeling, CVAT, etc. But for this study CVAT (computer vision annotation tool), an open-source image annotation tool developed by Intel Corporation was used. Figure. 7.3. shows the interface of CVAT image annotation tool.

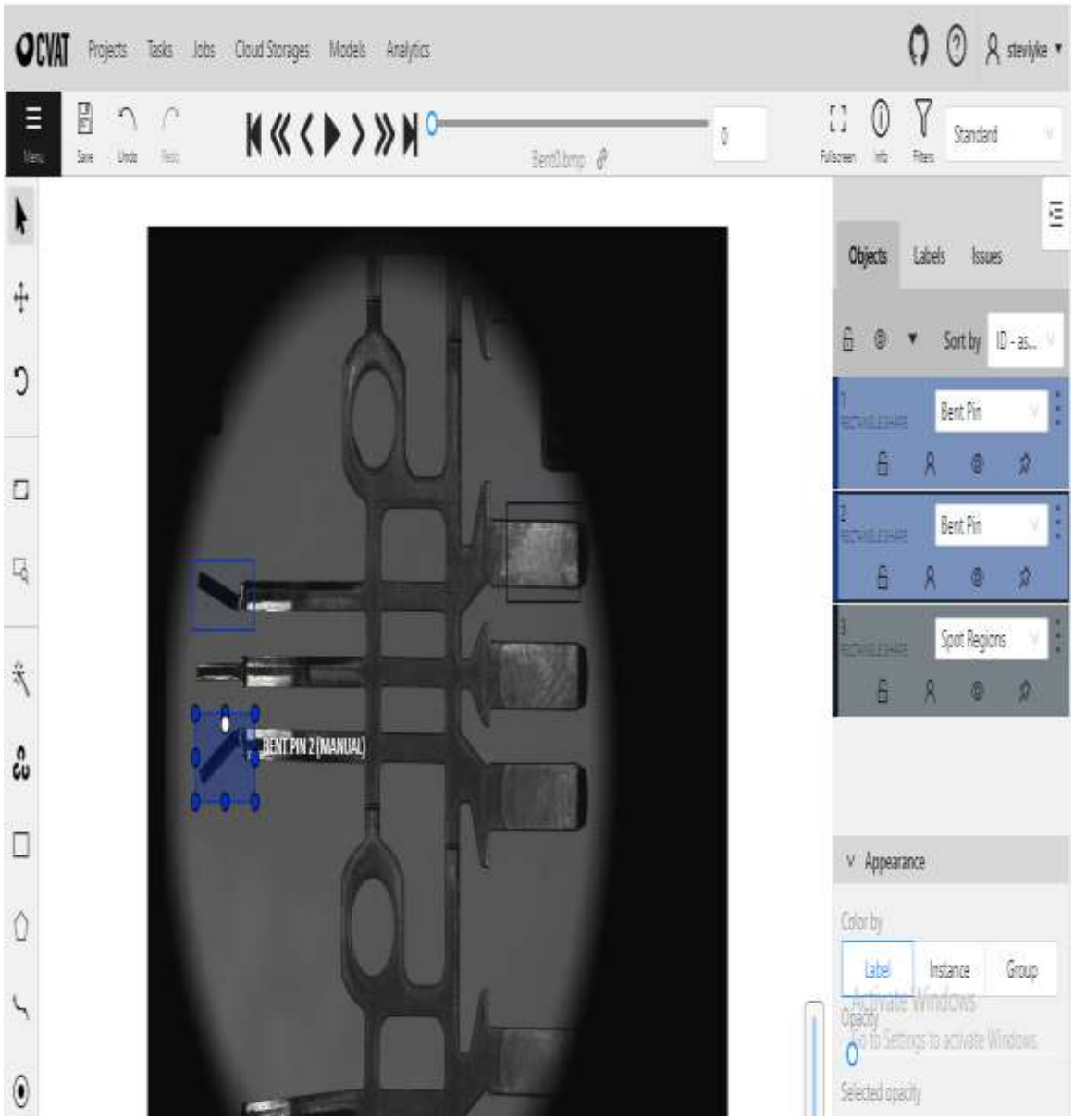


Figure 7.3. Interface of CVAT image annotation tool.

7.3. Proposed YOLO V3 Network

Defects on steel terminal surfaces are the center-point of this research work, and the size of the defect images can be categorized as ranging between medium and large. YOLO v3 was trained on the COCO dataset comprising of 80 classes and K-Means clustering was used to obtain 9 size prior anchors, thereby making it most suited for the subject of this research project. Multi-scale prediction method is an integral part of the YOLO v3 [20] network aim to optimize the detection of targets of different sizes and shapes. Hence for this research YOLO v3 is modified at the Convolution layers before each of the three (3) YOLO layers. The custom configurations are geared at aligning the feature map depth for the last convolutional layers with the custom dataset used in

this study before feeding its output to the YOLO layer, by modifying the filter and class parameters. This is necessary as the COCO dataset on which YOLO v3 was originally trained has a default depth of 255 and a total of 80 classes.

7.3.1. Proposed YOLO V3 Anchor Boxes for Cluster Analysis

It is noted that Faster RCNN uses anchor box method just like YOLO v3. Anchor boxes are a primary set of priori candidate boxes with fixed width and height, and whose size and proportion impacts the obtainable accuracy and speed of the network in target detection tasks. K-means clustering is utilized for determining the size and proportion, and for instance, the COCO dataset, about 9 sizes of anchor boxes are clustered as shown in Table 6.3. For the dataset used in this study, the output shape tensors are (13 x 13 x 33), (26 x 26 x 33), and (52 x 52 x 33), while the cluster centers remain as shown in Table 6.3. The original Yolo v3 network has output tensors of (13 x 13 x 255), (26 x 26 x 255) and, (52 x 52 x 255) [20].

YOLO v3 integrates the average IOU (Average Intersection over Union) as a criterion for the clustering result as long as the data set is clustered. The higher the average IOU, the better the derived clustering, and the average IOU for the dataset in this work ranged from 0.6632, 0.6454, to 79.19 which are > 0.5, indicating reliable clustering and target estimation mechanism.

Internally, YOLO v3 calculates average IOU using Formula (7.1) below,

$$\text{Average IOU}, F = \left(\sum_{i=1}^{k} \sum_{j=1}^{nk} I\ \text{IOU}\ (T, P) \right) \div n \qquad (7.1)$$

where T = real target box

P = center of the cluster

I_{IOU} (T, P) = IOU (intersection ration) between the target box and the cluster box

k = total number of clusters

n = total number of samples

nk = number of samples at the kth cluster center

i = sample number and,

j = sample number of the cluster center

K = number of clusters in the k-means clustering, which represents the number of clusters you want to get and you are free to choose the number.

7.3.2. Network Structure of Proposed YOLO V3

The optimized YOLO v3 network has the advantage of strong semantic information useful for category detection. The target location and category must first be determined in order to perform target detection, and this is where the relevance of the YOLO v3 network's multiscale fusion prediction method proves indispensable. The up-sampling mechanism merges the high-level network layers with the bottom-level layers of the network to get finer-grained feature information for positioning and classification, as shown in Figure 7.4, and Figure 7.5 shows the optimized network structure of the YOLO v3 network that is proposed in this study. The original YOLO v3 network features 6 DBL units with a 1x1 convolution in the target detection output layer. However, the proposed YOLO v3 algorithm is further optimized by adding a residue block to extract fine-grained deep steel surface defect features with a 4-scale feature map for object detection, in contrast to the original YOLO v3 with a 3-scale feature map. This allows the deeper networks to extract more detailed features. In this study, to prevent the disappearance of the gradient by reusing extended features, 6 DBL units are converted into 4 DBL units and 4 res_unit.

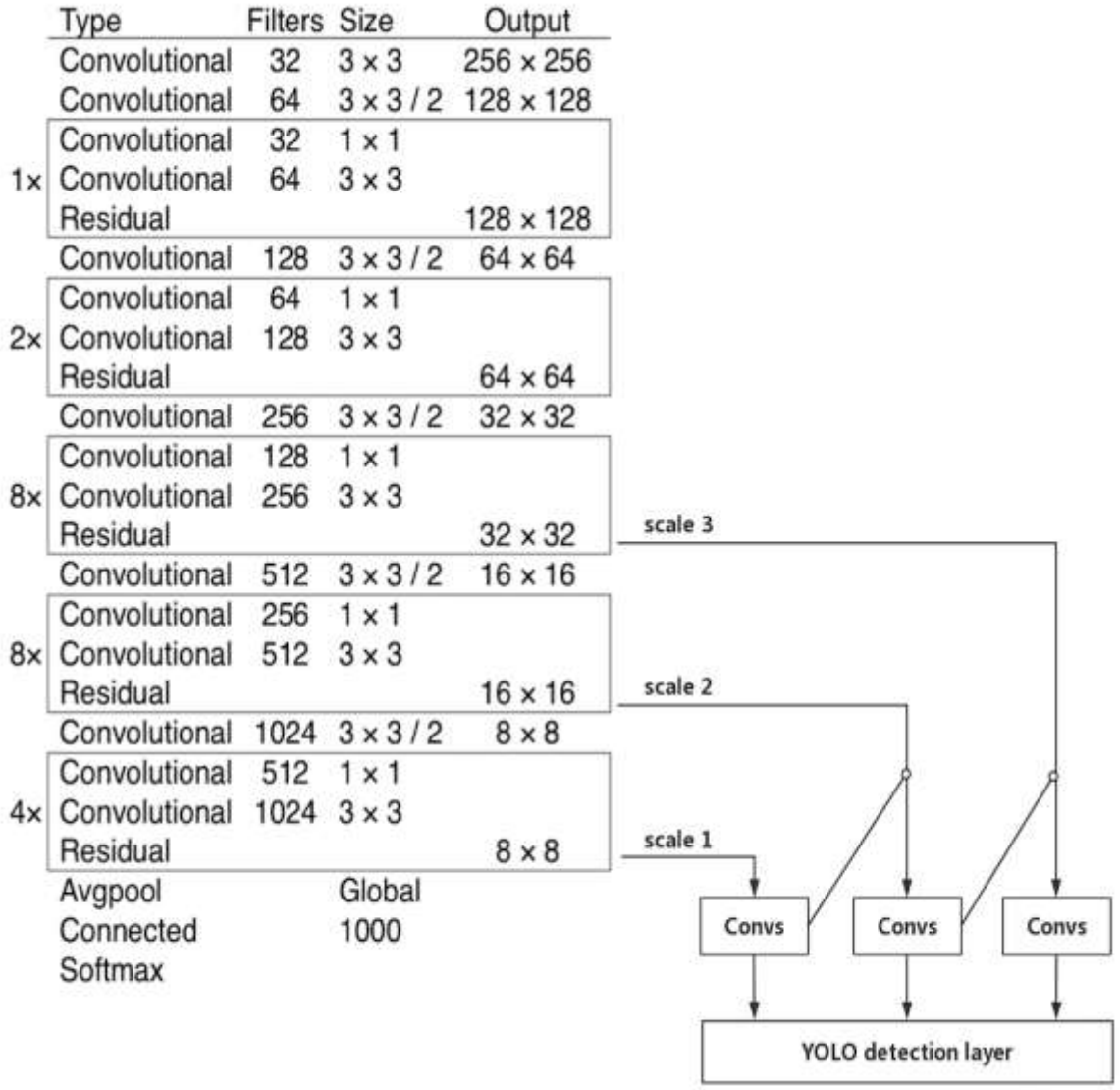


Figure 7.4. Network structure of the YOLO v3 network.

The model structure developed in this study, shown in Figure 7.5, is both optimal and suitable for the subject of this study well-tuned to deliver excellent performance in training and inferencing.

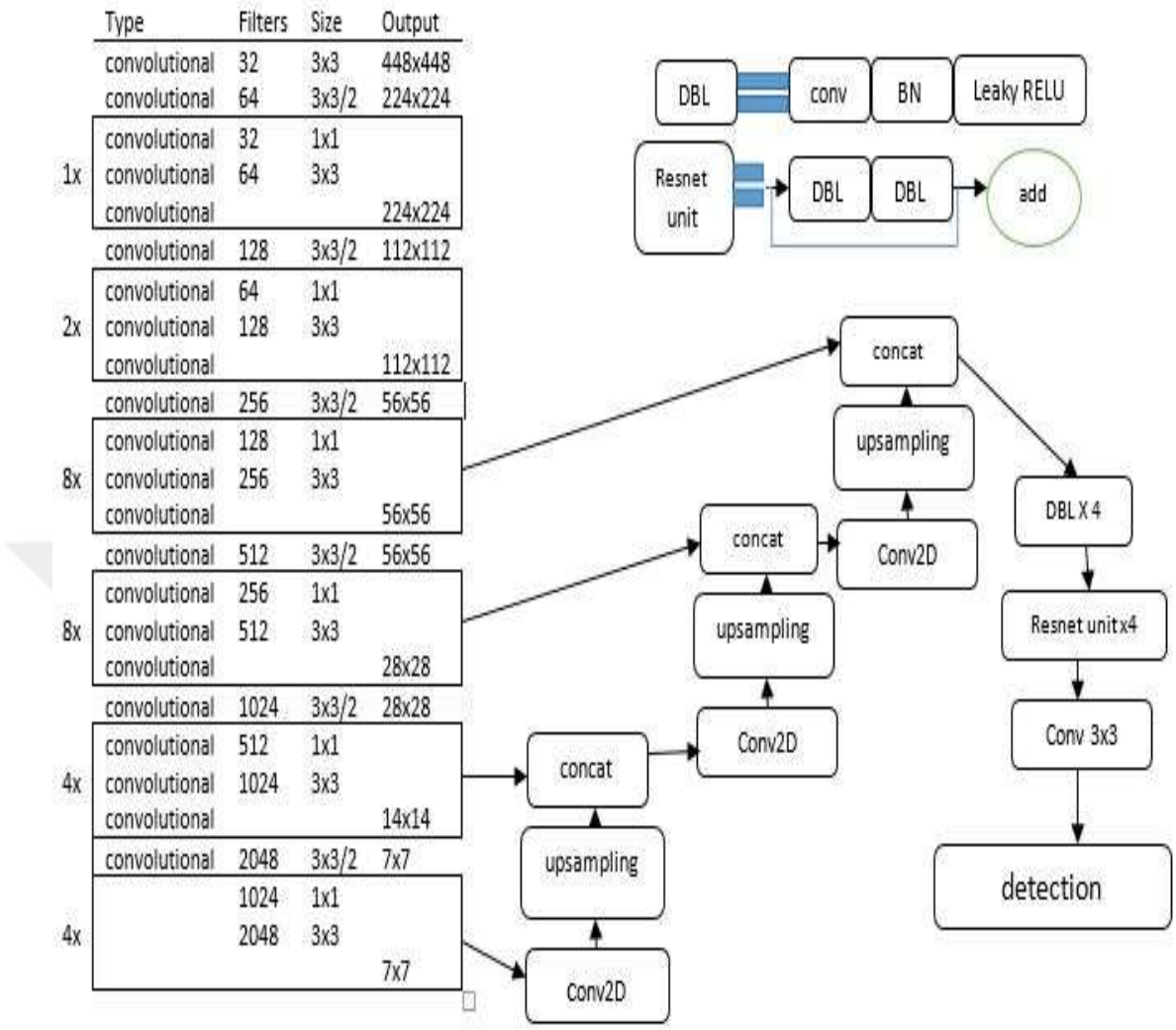


Figure 7.5. Network structure of proposed optimized YOLO v3 algorithm

7.3.3. Model Training of Proposed YOLO V3 Network

To kick-start YOLO v3 network training, the Darknet-53 [33] backbone architecture for YOLO v3 was installed on the Jetson Nano and the configurations for the custom dataset was tweaked to meet the requirements of the subject of this study.

For training, a total of 1200 images representing 80% of the total images was used, while 225 images were used for validation and 75 were reserved for testing. This choice of train_validation_test split was arrived at due to limited image datasets to make more data available for training, which is very important. In-memory, a 4-stage data augmentation was applied to compensate for the small dataset – hue, saturation, exposure, and angle manipulations were applied. The weight drop was set to 0.0005 to reduce weights to avoid very large values and ensure network

stability. Momentum was set 0.9 to make the gradient more stable and is calculated thus in Formula (7.2).

$$\text{Momentum, m} = \Big(\text{m x (p)} + \big((1 - \text{m}) \text{ x c}\big)\Big) \tag{7.2}$$

Where m is the momentum, p is the previous gradient, and c is the gradient of current batch

The network training learning rate is set as 0.001, number of iterations (epoch) is 1200 with a graduated policy, the maximum batch count (max_batch) is 1200, total iterations is 1200 with policy steps of 960 (80% of max_batch) and 1080 (90% of max_batch).

As a result of the nature of object detection problems, model training and validation accuracy and loss are not very good metrics for evaluating the performance of a network and are rarely used even in the original Yolo paper, instead Mean Average Precision (mAP), average loss, and average IOU (average IOU) are used. The mAP compares the ground truth bounding box to the detected box and returns a score. The higher the score, the more accurate the model is in its detections [32].

Network training lasted for approximately 12 hours due to memory and CPU bottlenecks due to the fact that the Jetson hardware used in the study are a lower model with a minimal GPU and memory allotment capacity. During training, the model training performance was evaluated by activating the mean average precision and average loss dynamic calculation over the course of the entire 1200 iterations. Sigmoid function was used for batch normalization to optimize the prediction probabilities of the model to suit the requirements of multi-class detection and classification. The network architecture of the YOLO v3 model proposed in this study is shown in Figure 7.5.

8. RESULTS AND DISCUSSION

The experimental results and accompanying discussions are contained in this section.

8.1. Optimized YOLO V3 Training Results

At the end of training, the YOLO v3 model trained with the dataset in this study, an mAP of 68% was achieved after 200 iterations at 50% IOU Threshold (mAP@0.50 = 68%). At 1200 iterations (end of training), the best mAP of 97.31% was achieved, while Precision = 0.93, Recall = 0.74, f1-score = 0.82, and Average IOU = 0.66, with an average detection time of 15 seconds. The f1 score metric evaluates the balance between the precision and recall. If the value of f1-score is high, that implies both precision and recall are high. A low f1-score imply undoubtedly that there is a high imbalance between the precision and recall [32]. The experimental value for f1-score (0.82) obtained at the end of the network training indicates a small imbalance between precision and recall, therefore the confidence in the network's target prediction is reliable [35]. Equation (8.1) is used in calculating f1-score.

$$F1 - \text{Score} = 2\{(P \times R) \div (P + R)\} \tag{8.1}$$

Where P and R represent precision and recall, respectively.

From the mAP values and the average loss values (curve) it is clear that the results of the optimized YOLO v3 network obtained herein at the end of the model training are convincingly ideal, acceptable and can be adjudged as a qualified network. The average precision values (AP) for each of the classes in the dataset used in network training also indicate excellent performance.
The per-class average precision, AP, training performance of the optimized YOLO v3 network after 1200 iterations is shown in Figure. 8.1.and Table 8.1. show the overall results.

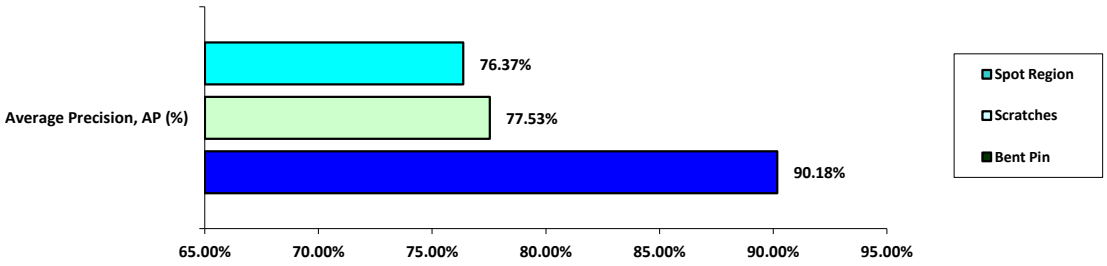


Figure 8.1. Per-class average precision at end of model training.

Table 8.1. Training performance results of proposed YOLO v3.

		AP_{50} (%)	Precision	Recall	F1-score	IOU	mAP@50 (%)
Iterations	**Class**						
	Bent Pin	90.18					
1200			0.93	0.74	0.82	0.6624	81.31
	Scratches	77.53					
	Spot Regions	76.37					

Object detection networks are mostly tested for validity and suitability using Precision, P and Recall rate, metrics. Precision indicates how confident the model is in classifying a sample as positive, while recall indicates the number of negative targets the model classified as positive. The precision rate, P, and Recall rate, R can be calculated using Formula (8.2) and (8.3).

$$\text{Precision,} \quad P = \frac{N_{TP}}{N_{TP} + N_{FP}} \tag{8.2}$$

$$\text{Recall,} \quad R = \frac{N_{TP}}{N_{TP}+N_{FN}} \tag{8.3}$$

Where,

N_{TP} = the number of targets detected correctly (True Positive) by the network,

N_{FN} = the number of targets not recognized by the network (false negatives) and

N_{FP} = the number of targets incorrectly recognized by the network (false positives).

The higher the precision, the more confident the model is in classifying a sample as positive. The higher the recall, the more positive samples the model correctly classifies as positive. The F1 Score metric measures the balance between precision and recall. If the value of F1 score is high, it means that both precision and recall are high. A lower F1 value means a larger imbalance between precision and recall [32].

8.1.1. Training Performance Comparative Analysis

This study will attempt to compare the model proposed in this study with other models on mostly hot-rolled steel strips available in the literature.

Table 8.2. shows the defects detection performance metrics (mAP) comparison of the optimized YOLO v3 model proposed in this study with other models, on steel surfaces.

Table 8.2. Comparative analysis of proposed model performance.

Model	Performance Metric			
	Anchor Boxes	**IOU**	**mAP (%)**	**Detection Time (ms)**
Faster R-CNN	9	62.33	91.2	1268
ANN	-	-	83.44*	2434.24
KNN	-	-	75.27*	2896
SSD	9+	-	82.7	
ORIGINAL YOLO V3	9	0.56	57.9	51
OPTIMIZED YOLO V3	6	0.66	97.31	65

*models with no mAP but rather accuracy scores

The results obtained during training indicate that the optimized YOLO v3 network proposed in this study performs comparably with the SSD and lower than Faster R-CNN in terms of mean average precision (mAP) in target detection, including the original YOLO v3 network that was trained with the COCO dataset. However, it must be pointed out that models like Faster R-CNN and SSD are generally more suited for larger targets and not small targets like surface defects like the type that is the subject of this study. In terms of detection time, the model proposed in this study outperforms all other models in literature unless the original YOLO v3 as a result of difference in hardware platform in terms of GPU and memory capacity. Both the original YOLO v3 and the other models were trained on an M40/Titan X GPU which is over 10x faster than the Jetson Tegra X1 used in this study. This implies that the model implementation proposed in this study is reliable and qualifies as a suitable network for implementing defect detection The precision, recall and F1 scores obtained at the end of the training also confirm the robustness and prediction confidence of the YOLO v3 network proposed in this study. Also, the model proposed in this study shows a more enhanced and balanced performance when compared with other target detection models, as can be seen from experimental results related to IOU and F1 score.

8.2. Test Performance Results

Table 8.3. shows the model performance during testing with image data the model had never seen before. Figure. 8.2 show a bent pin defect prediction output, Figure. 8.3 shows a scratches defect prediction output, and Figure. 8.4. shows a spot region defect prediction output, respectively.

Table 8.3. Test performance of proposed model.

Class	Prediction Confidence (%)
Bent pin	96
Scratches	79
Spot region	72

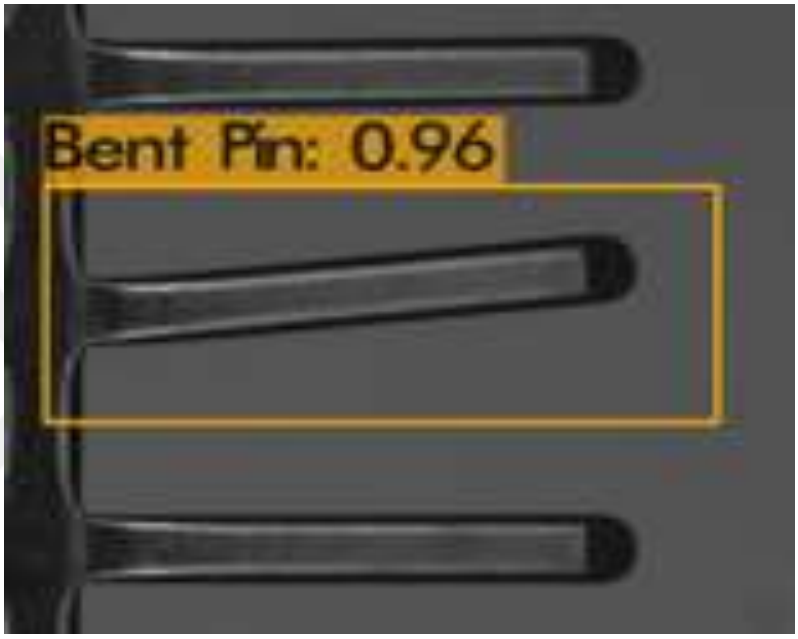


Figure 8.2. Bent pin defect prediction output.

Figure 8.3. Scratches defect prediction output.

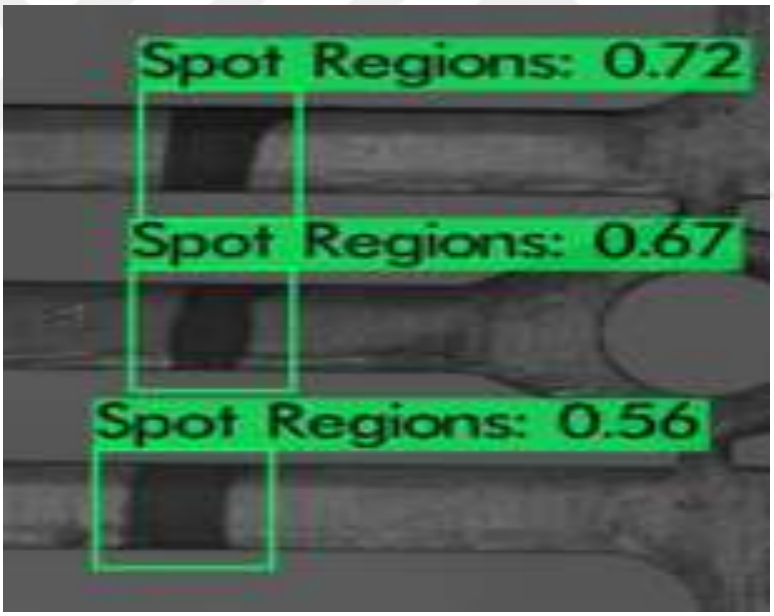


Figure 8.4. Spot regions defect prediction output.

The results obtained during testing by feeding the model with image data it had never seen before show that the model performed better during testing as the average prediction confidence of the model (mAP) is approximately 83% higher than the 81.31% obtained during training. This indicates the models inferencing quality is appropriate and there was no overfitting over the training data.

8.3. Overall Model Results Analysis

The experimental results of the model proposed in this study indicate that it is a qualified network for target detection tasks, especially for small targets like surface defects. The test results most especially, shows that the model prediction confidence is idea and can detect even very subtle and multi-surface defects on steel terminals without difficulty, with the added advantage of speed and accuracy. This underscores the inferencing capabilities of the proposed model and its unique suitability for wider implementation in steel terminal manufacturing industries.

9. CONCLUSION

The optimized model herein proposed is primarily centered on an optimized YOLO v3 network, known for its ability to select anchor boxes, perform cluster analysis of custom datasets, and calculate appropriate numbers and sizes of candidate boxes. The backbone network anchor boxes are redesigned in this document to have 6 anchor boxes instead of the 9 anchor boxes in the backbone network. This is to ensure that the proposed algorithm can detect small and very small targets and also perform multiple target detection without compromising the reliability of the detection.

In the proposed YOLO v3 algorithm, a sigmoid function was used instead of the regression method like the softmax function. This is owing to the fact that the softmax method has the characteristic of increasing the likelihood of the most likely category while suppressing the value of other categories, which is not idea in multi-class detection and classification problems, which forms the subject of this study.

Through experiments, it can be seen that the algorithm proposed in this study has highly balanced and stable performance when compared to other target detection algorithms, especially when it comes to small targets specifically. The recall, precision, f1-score and mAP of the algorithm proposed in this study have been significantly improved compared to the original YOLO v3 network. However, per the detection speed, the proposed algorithm is slower than the original YOLO v3, which is attributed to the low computational power of the hardware platform used in this study. Balanced detection performance, stability, and speed, these performance metrics meet the industrial sector's needs for near real-time accuracy and detection of surface defects on steel terminals. The detection speed of the YOLO v3 algorithm herein proposed is about 5% of Faster R-CNN.

YOLO v3 shows promising performance as against other models, although there is still enormous room for improvement in the areas of network performance relating to detection accuracy and speed, and this should be the focus of the next research directions. A diluted flowchart of the overall procedures and phases in this work is shown in Figure 9.1.

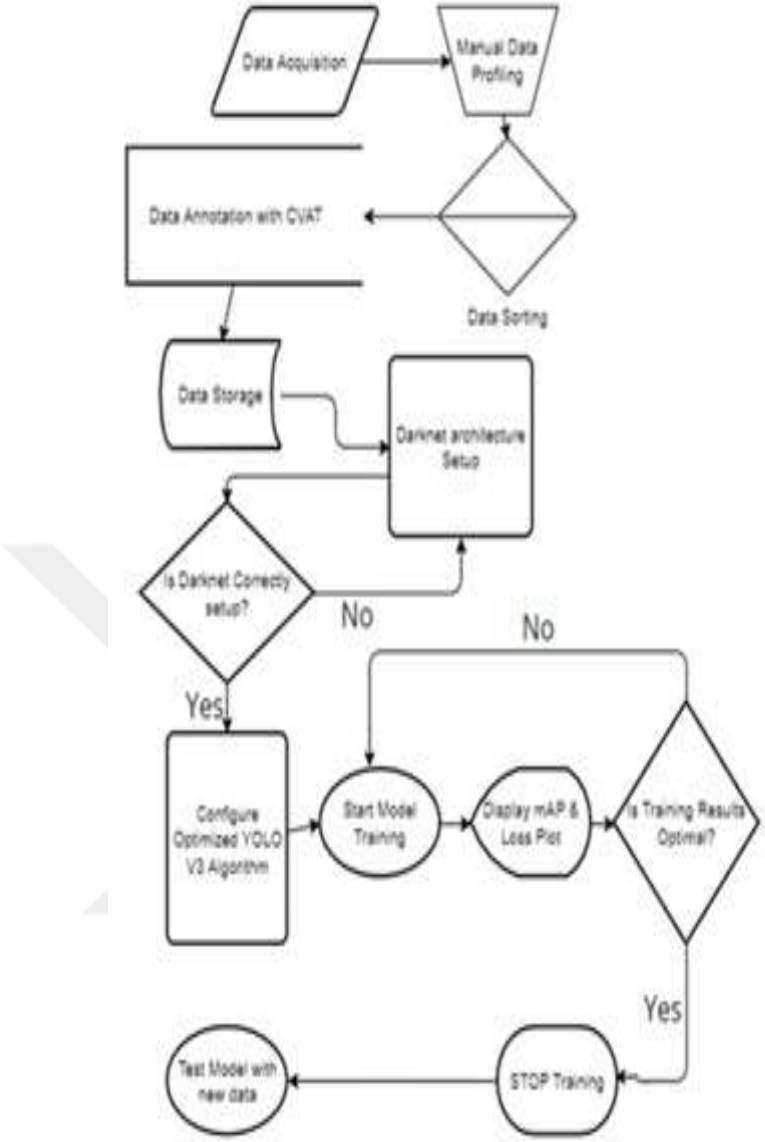


Figure 9.1. Overall Flowchart of processes followed in the study.

RECOMMENDATİONS

From experiments and research, the author in this study has a few observations that will form the fulcrum of his recommendations. One, the author recommends that a broader and expansive research project be commissioned by the university with focus to acquire massive amounts of industry-grade data from live production lines like Hatko's. This will make more data available for a more in-depth algorithm design and development.

Two, a more expansive laboratory facility in terms of hardware with more rugged GPU, TPU, and CPU capacity be provisioned by the university and its partner industries to make it easier for researchers to have access to the right tools and resources for experimental work and design.

Three, research projects with far-reaching industrial implications and usefulness should be organized into research teams and groups with each team member focusing on a specific aspect of the project like data preparation (annotation), algorithm design, algorithm testing, software engineering (app design for model deployment and management). Focus is key in research projects as having to do many things by a single researcher distracts for the core nitty-gritties of scientific research work. Also, industry focused research studies with support from industries or research bodies should come with support for the researchers working on the studies, not just providing of equipment.

Lastly, I recommend that the next phase of the study be commissioned by the university to continue working on improving the results of this study in terms of model accuracy, speed, software deployment and also data maintenance and storage so that the university can become a hub of academic researchers looking for industry-grade and benchmark datasets for their work. Not only will this help the global ranking of universities, but it will also attract more global funding and partnership.

REFERENCES

[1] Thornton, M.; Han, L., and Shergold, M. (2012). Progress in NDT of resistance spot welding of aluminum using ultrasonic C-scan. NDT E Int. 2012, 48, 30–38.

[2] Song, K., and Yan, Y. (2013). A noise-robust method based on completed local binary patterns for hot-rolled steel strip surface defects, Applied Surface Science, vol. 285, pp. 858-864.

[3] Dai, J., Qi, H., Xiong, Y., Li, Y., Zhang, G., Hu, H., and Wei, Y. (2017). Deformable convolutional networks. In: International Conference on Computer Vision, pp. 764–773.

[4] Tamas C., and Gastone C. (2020). Visual-Based Defect Detection and Classification Approaches for Industrial Applications—A SURVEY, March 2020.

[5] Yang, Y., Pan, L., Ma, J.; Yang, R., Zhu, Y., Yang, Y., and Zhang, L. (2020). A High-Performance Deep Learning Algorithm for the Automated Optical Inspection of Laser Welding. Appl. Sci. 2020, 10, 933.

[6] Aditya M., and Ali A. (2020). One-Shot Recognition of Manufacturing Defects in Steel. Prodia Manufacturing 48 (2020) 1064 – 1071.

[7] Simonyan K., and Zisserman A. (2015). Very Deep Convolutional Networks for Large Scale Image Recognition. In: International Conference on Learning Representations (ICLR); 2015.

[8] NEU surface defect database http://faculty.neu.edu.cn/yunhyan/NEU_surface_defect_database.html/.

[9] Cao X, Yang J, Zhang J, Wang Q, Yap P-T, and Shen D. (2018). Deformable image registration using a cue-aware deep regression network. IEEE Trans Bio-med Eng. 2018; 65:1900–11.

[10] Shao, C., Xu, G., Xu, H. (2015). Research of Recognition Method for Surface Defects of Hot-rolled Round Steel Based on Image Processing, Advanced Materials Research, vol. 1090, pp.84-89.

[11] Neogi, N., Mohanta, D., and Dutta, P. (2014). Review of vision-based steel surface inspection systems, Eurasip Journal on Image and Video Processing, vol. 50, pp.1-19.

[12] He, K., Zhang, X., Ren, S., Sun, J. (2016). Deep residual learning for image recognition. In: Conference on Computer Vision and Pattern Recognition, pp. 770–778.

[13] Lv, X., Duan, F., Jiang, J., Fu, X., and Gan, L. (2020). Deep active learning for surface defect detection. Sensors, 20(6), 1650.

[14] Zhou S, Chen Y, Zhang D, Xie J, and Zhou Y. (2017). Classification of surface defects on steel sheet using convolutional neural networks. Material Tehnol 2017; 51:123–31.

[15] Choi DC, Jeon YJ, Kim SH, Moon S, Yun JP, and Kim SW. (2017). Detection of pinholes in steel slabs using Gabor Filter combination and morphological Features. ISIJ Int. 2017; 57:1045–53.

[16] Yun, J. P., Choi, S., Kim, J., and Kim, S. W. (2009). Automatic detection of cracks in raw steel block using gabor filter optimized by univariate dynamic encoding algorithm for searches (udeas). Ndt & E International, 42(5), 389–397.

[17] Zhang, J., Li, H., Yang, B., Wu, B., and Zhu, S. (2020). Fatigue properties and fatigue strength evaluation of railway axle steel: Effect of micro-shot peening and artificial defect. International Journal of Fatigue, 132, 105,379.

[18] Zhu X, Liu Y, Qin Z, and Li J. Emotion classification with data augmentation using generative adversarial networks. 2017 [Online]. Available: https://arxiv.org/pdf/1711.00648.

[19] Sohn K, Lee H, and Yan X. (2015). Learning structured output representation using deep conditional generative models. in Advances in Neural Information Processing Systems 2015:3483–91.

[20] Tamas C. and Gastone C. (2020). Visual-Based Defect Detection and Classification Approaches for Industrial Applications—A SURVEY.

[21] Caleb, P., and Steuer, M. (2010). Classification of surface defects on hot rolled steel using adaptive learning methods, Fourth International Conference on Knowledge-Based Intelligent Engineering Systems and Allied Technologies, Bright, UK.

[22] Pakkanen, J., Iivarinen, J., Rautkorpi, R., and Rauhamaa, J. (2014). Content-based retrieval of surface defect images with PicSOM, International Journal of Fuzzy Systems, vol. 6, no. 3, pp.160-167, 2004.

[23] Hongbin, J., Yi, L., and Jianjun, S. (2004). An Intelligent Real-time Vision System for Surface Defect Detection, 17th International Conference on Pattern Recognition, Cambridge, UK, August 23-26.

[24] Keesug, C., Kyungmo, K., Lee, J. (2006). Development of defect classification algorithm for POSCO rolling strip surface inspection system, SICE-ICASE International Joint Conference, Busan, Korea, Oct. 18-21, 2006.

[25] Smriti, H. and Bhandari, S. (2008). A Simple Approach to Surface Defect Detection, the Third International Conference on Industrial and Information Systems, Kharagpur, India.

[26] Ren, S., He, K., Girshick, and Sun. J. (2017). Faster R-CNN: Towards real-time object detection with region proposal networks. IEEE Trans. Pattern Anal. Mach. Intel., vol.39, no. 6, pp. 1137-1149.

[27] Ehab, A., Samy H., and Nashat, A. (2017). Detection of Steel Surface Defect Based on Machine Learning Using Deep Auto-encoder Network. Available: http://ieomsociety.org/ieom2017/papers/57.pdf.

[28] Aditya M., Ali A. (2020). One-Shot Recognition of Manufacturing Defects in Steel. Prodia Manufacturing 48 (2020) 1064 – 1071.

[29] Intel Corporation (2018). available: https://www.intel.com/content/www/us/en/developer/articles/technical/use-machine-learning-to-detect-defects-on-the-steel-surface.html.

[30] Viraktamath. S, Madhuri. Y, and Rachita B. (2021). Object Detection and Classification using YOLOv3. International Journal of Engineering Research & Technology (IJERT). Vol. 10 Issue 02, February-2021.

[31] Redmon J. and Farhadi A. (2015). YOLOv3: An incremental improvement. in proc. IEEE Conf. Computer Vision Pattern Recognition, Jun. 2015, pp. 779-788.

[32] Karlijn Alderliesten. (2020). YOLOv3 — Real-time object detection. May 28 2020.

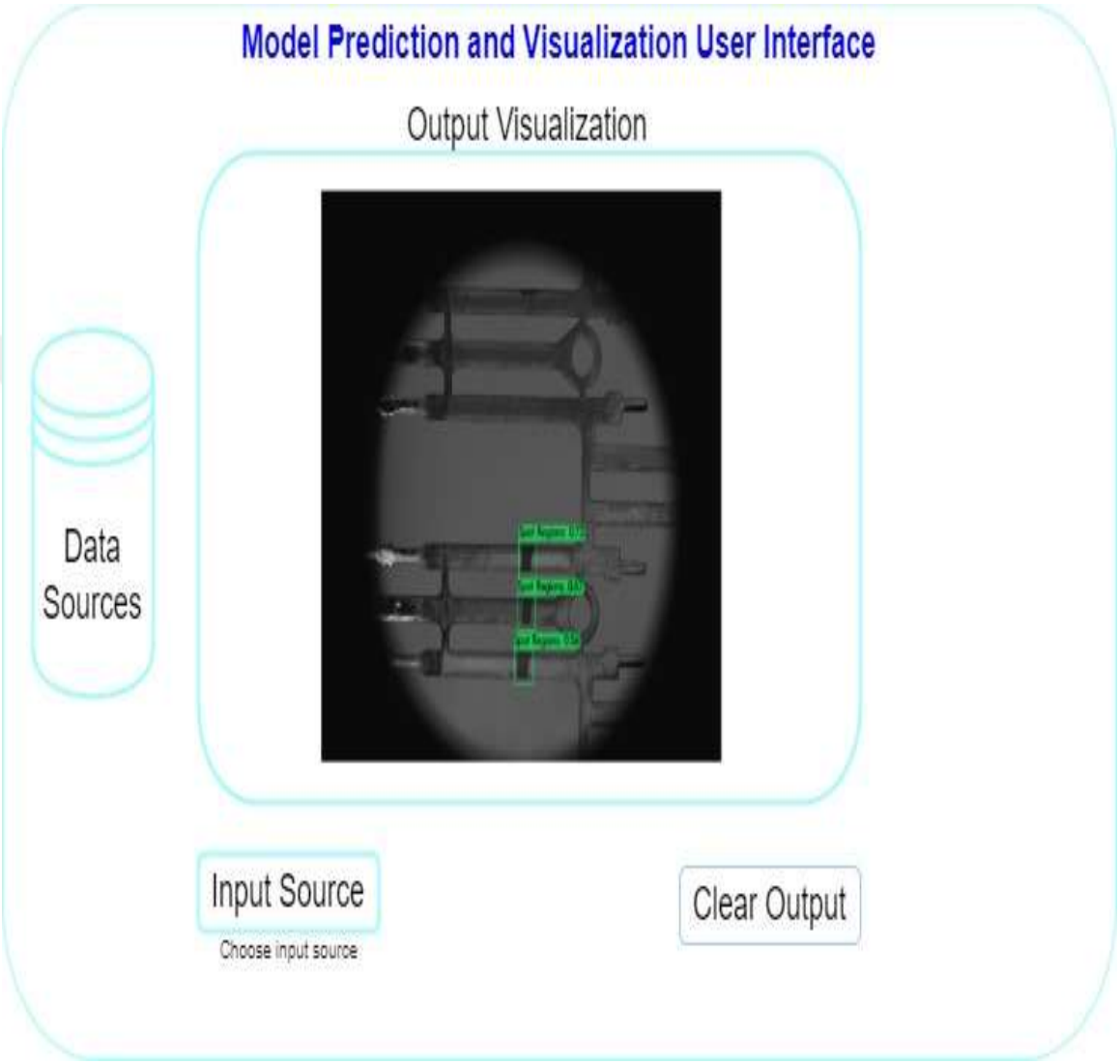
[33] Redmon J. (2016). Darknet: Open-source neural networks in c. http://pjreddie.com/darknet/.

[34] Preisser, J.S, Das K., Benecha, H. and J. W. Stamm. (2016). Logistic regression for dichotomized counts, Stat. Methods Med. Res., vol. 25, no. 6, pp. 3038-3056.

[35] Ahmad, F. (2021). Mean Average Precision in c. https://blog.paperspace.com/mean-average-precision/.

APPENDICES

APPENDIX-1: PROPOSED USER INTERFACE (GUI) FOR OPERATING THE MODEL IN A POSSIBLE DEPLOYMENT ENVIRONMENT.



CURRICULUM VITAE

Stephen Ikechukwu OKO-EGWU

PERSONAL INFORMATION

[redacted]

ACADEMIC ACTIVITIES

Paper:

1. Stephen I., and Erhan A. (2022). High Performance Network for Detection of Surface Defects on Hot-Rolled Steel Strips Based on an Optimized Yolo V3, the 2022 9th International Conference on Electrical Electronics Engineering (CETA 2022), March 2022.
2. Stephen I., Erhan A, and Ahmet T. (2022). Optimized Yolo v3 Based Defect Detection on Copper Alloy and Steel Terminals using Computer Vision, IEEE Transactions on Industrial Informatics (Under Review).