



Advanced Industrial Defect Detection using Optuna-Optimized YOLOv8 Architecture with Tiling Inference

by

Sandeep Kumar
(Student Number: STU230944)

An Advanced Computing Project submitted to the faculty of
Arden University Berlin, Germany

in accordance with the University's Ethics Guidelines and
in partial fulfillment of the requirements for the degree of

Master of Science
Faculty of Science, Technology, Engineering and Math (STEM)

Supervisor:	Zingsho Vashum
University:	Arden University Berlin
Submission Date:	21 January 2026
Word Count (Body):	8,656
Word Count (Headers):	333
Word Count (Captions):	40
Grand Total:	9,029

Abstract

Industrial surface defect detection is an important need in current industrial quality control processes. A complete system for detecting and categorizing 6 different kinds of metal surface defects has been developed by this research project as part of the YOLOv8s object detection architecture. Additionally, we propose a two-stage optimization approach that uses Bayesian Optimization with the Optuna framework to circumvent the constraints imposed by default hyperparameter settings. We also introduced a customized sliding window tiling technique to address high-resolution images typical to industrial environments while retaining all pixel detail. Trial 20 was our top performing model that had a mAP@50 of 0.985 and a mAP@50-95 of 0.827. The results show the reliability of our system over a range of morphological characteristics and illustrate the value of Bayesian Optimization for use in industrial deep learning systems.

Contents

1	Introduction	4
2	Literature Review: The Evolution of Automated Surface Defect Detection	6
2.1	Introduction	6
2.2	Traditional Computer Vision Methods	7
2.2.1	Statistical Texture Analysis	7
2.2.2	Spectral and Frequency Domain Methods	7
2.3	Mechanisms of Deep Learning in Inspection	7
2.3.1	CNN Building Blocks	7
2.3.2	The Architectural Divide: Two-Stage vs. One-Stage	8
2.3.3	Beyond Convolutions: The Vision Transformer (ViT) Paradigm	8
2.4	Evolution of the YOLO Architecture	9
2.4.1	The early stages (v1-v3)	9
2.4.2	Optimizations (v4-v5)	9
2.4.3	Modern and state-of-the-art (v8-v11)	9
2.5	Key mechanisms in industrial detection	10
2.5.1	Enhanced attention mechanisms	10
2.5.2	Path aggregation	10
2.6	Data-Centric AI and Anomaly Detection	11
2.6.1	Benchmarks Datasets	11
2.6.2	Generative AI for data augmentation	11
2.6.3	The Unsupervised Alternative: Anomaly Detection	11
2.7	Conclusion	12
3	Justification for Choosing YOLOv8	13
3.1	Justification for Choosing YOLOv8, As Opposed to More Recent Versions	13
3.1.1	Model Ecosystem and Hardware Support	13

3.1.2	Tradeoffs in Newer Architectures	14
3.1.3	Standardization to Facilitate Comparative Analysis	14
4	Theory	15
4.1	An Overview of AOI and One-Stage Detector Evolutions	15
4.1.1	YOLO Architectural Developments for Use in Industry	16
4.1.2	The Specialized Challenges of Defect Detection	16
4.2	The Backbone: Feature Extraction Mechanics	17
4.2.1	The C2f Module: Optimizing Gradient Flow	17
4.2.2	Convolutional Modifications: From 6x6 to 3x3 Kernels	18
4.2.3	Spatial Pyramid Pooling Fast (SPPF): Increasing the Receptive Field	19
4.2.4	Lightweight Variants: GhostNet	19
4.3	The Neck: Path Aggregation and Feature Fusion	20
4.3.1	FPN and PANet Mechanics	20
4.3.2	Dealing with Multi-Scale Defects	21
4.4	The Head: Decoupled and Anchor-Free Detection	21
4.4.1	The Decoupled Head Architecture	22
4.4.2	Anchor-Free Detection	22
4.5	Label Assignment	23
4.5.1	The Transition from Static to Dynamic Label Assignment	23
4.5.2	Mathematical Description of TAL	23
4.6	Training Losses and Optimizer	24
4.6.1	Binary Cross Entropy (BCE) Loss Function for Classification . . .	25
4.6.2	Complete Intersection Over Union (CIoU) Loss Function for Localization	25
4.6.3	Distribution Focal Loss (DFL):	26
4.7	Augmentation and Model Scaling	27
4.7.1	Mosaic Augmentation	27
4.7.2	Mixup Augmentation	28
4.7.3	Model Scaling Parameters	28
4.7.4	Flops vs. Performance Trade Off	29
4.8	Domain-Specific Application Analysis	29
4.8.1	Steel Surface Defect Detection	29
4.8.2	Photovoltaic (PV) Cell Inspection	30
4.8.3	Fabric and PCB Inspection	30
4.9	Conclusion	30
5	Optimization and Training Methodology	32

5.1	Phase 1: Bayesian Hyperparameter Search	32
5.2	Phase 2: High-Resolution Fine-Tuning	33
5.3	Hyperparameter Evolution	33
5.4	Computational Efficiency	33
6	Results and Performance Evaluation	35
6.1	Quantitative Metrics	35
6.2	Class-wise analysis	36
6.3	Stability of convergence	36
6.4	Comparison with Other Models	36
6.5	Qualitative Images	36
7	Discussion: Shortcomings and Practical Challenges	39
7.1	Shortcomings and Possible Failure Modes	39
7.2	Hardware Limitations	40
7.3	Ethical and Social Aspects	40
8	Conclusion and Future Research	41
8.1	Summary of Contributions	41
8.2	Directions for Future Research	41

Chapter 1

Introduction

Industry 4.0 has revolutionized how manufacturing and quality control operate. The primary way that humans have inspected products has been by manually inspecting the product's surface for flaws. Although the best inspectors will never completely eliminate all error due to the physical limitations of the human eye (including fatigue, cognitive bias, and the inability to visually inspect everything in every inspection) human inspectors are still the primary method used to inspect surfaces for flaws today. As manufacturing continues to move at a faster pace, it will become increasingly necessary for machines to take over the role of inspectors because the speed of manufacturing far exceeds the speed at which a human inspector can visually inspect a product.

This thesis aims to develop a fully-automated surface defect detection system utilizing the You Only Look Once version 8 (YOLOv8) architecture. AOI (Automated Optical Inspection) is a critical part of modern smart factories and can detect and classify defects quickly and accurately. However, industrial surfaces present unique challenges for computer vision. These include defect patterns that have low contrast against background, defects that vary in size and shape, and defects that occur under variable lighting conditions.

In addition to the issues described above, this research identifies six different types of defect classes that were identified and included in the research: crazing, inclusion, patches, pitted surface, rolled-in scale, and scratches. All of these are unique in their morphology; e.g., "crazing" refers to a network of fine hair-like cracks that can be difficult to identify in low resolution images and may appear to be similar to surface texture. An "inclusion" refers to foreign matter that has been embedded in the product during manufacturing.

An "inclusion" may appear as a black spot and can be irregularly shaped. Any successful system must be able to detect and classify all of these defect classes in spite of possible environmental contamination.

One of the goals of this research is to achieve "Zero Defect Manufacturing" (ZDM). ZDM is a goal of manufacturers who want to produce defect-free parts. To achieve ZDM, simply detecting defects is insufficient. Rather than relying solely on detecting defects, a more comprehensive framework of continuous optimization of the detection and classification processes must be developed.

To develop this framework of continuous optimization, this thesis develops a two-phase hyperparameter optimization strategy using Optuna, a Bayesian optimization framework. Through separating the optimization process into a coarse global search and a high-resolution local refinement (fine-tuning), we demonstrated that common performance plateaus can be overcome. Furthermore, we demonstrated that combining architectural efficiency with advanced data engineering techniques (i.e., tiling strategies and robust augmentation methods such as Mosaic and Mixup) are effective means of achieving optimal performance.

Following this introduction, the remainder of this document is organized along the research cycle. Section 2 reviews the relevant literature regarding the history of object detection, from early classical algorithms to state-of-the-art deep learning architectures. Section 3 examines the theoretical foundations of YOLOv8 and describes the backbones, necks, and heads of the YOLOv8 architecture. Section 4 describes the development of the proposed system design and describes the custom tiling strategy that was developed to allow processing of large-scale high-resolution image data sets. Section 5 describes the experimental methodology that was employed to develop and test the proposed system and describes the Optuna-based optimization process. Section 6 presents the analysis of the results of testing the proposed system, and highlights the success of our "Trial 20" model. Finally, Section 7 describes some of the limitations of the proposed approach and proposes several potential areas for further research.

Upon completion of this work, we anticipate providing a scalable blueprint for implementing deep-learning based defect detection systems that will meet both the accuracy and real-time requirements of modern industrial quality control.

Chapter 2

Literature Review: The Evolution of Automated Surface Defect Detection

2.1 Introduction

The industrial manufacturing sector is currently undergoing a significant transformation as part of the fourth Industrial Revolution, or Industry 4.0 [27]. A central component of this new paradigm is the concept of Zero Defects Manufacturing (ZDM) [45]. ZDM is designed to reduce waste and improve product quality through intelligent data-driven process control [43].

Automated Surface Inspection (ASI) Systems: ASI systems have progressed from being peripheral quality control mechanisms to being an integral part of the production process [5]. These systems identify, locate and categorize surface anomalies on various materials including hot rolled steel strip and semiconductor wafers [40]. In terms of financial implications, defects such as "rolled in scale" and "crazing" can be used as stress concentrations that result in catastrophic failure of structures [4].

In the past, manual inspection rarely achieved greater than 80% reliability due to fatigue and attentional drift [52]. Early automation systems that utilized Traditional Computer Vision (TCV) partially bridged the gap between manual inspection and reliable automated inspection [42]. TCV however, was "brittle" in its ability to deal with stochastic manufacturing variation [67]. Deep Learning (specifically Convolutional Neural Networks - CNNs) created a watershed moment in the history of computer vision, moving the focus from manual "feature engineering" to automatic "feature learning" [29].

2.2 Traditional Computer Vision Methods

Prior to the development of neural networks, defect detection relied heavily on statistical signal processing, assuming that a defect represented a local anomaly in an otherwise homogeneous texture [68].

2.2.1 Statistical Texture Analysis

Statistical Texture Analysis, particularly utilizing Gray-Level Co-occurrence Matrices (GLCM), was used to analyze continuous materials [17]. GLCMs represent the probability of the occurrence of combinations of pixel brightness values at specified spatial relationships [16]. Features such as Contrast (the amount of local variation), Homogeneity (the degree to which the distribution is close to diagonal), and Energy (textural uniformity) were used to describe a surface [59]. However, although these methods were efficient to compute, they were sensitive to orientation changes and insensitive to microscopic defects that may not affect the global statistics [54].

2.2.2 Spectral and Frequency Domain Methods

Frequency domain methods were developed to identify defects that manifested as high frequency interruptions [58].

Gabor Filters: Gabor filters use a combination of a Gaussian kernel and a sinusoidal function to enable the system to identify defects based on their directionality characteristics (i.e. scratches) [28].

Wavelet Transformations: Wavelets were also used to provide multi-resolution analysis, but unfortunately they were very sensitive to industrial noise (i.e. steam and oil mist) which often existed in the same frequency range as defects [12].

2.3 Mechanisms of Deep Learning in Inspection

CNNs have revolutionized inspection by allowing them to learn hierarchical feature representations directly from raw pixel data [71].

2.3.1 CNN Building Blocks

State of the art inspection focused CNNs are comprised of several key building blocks [18]:

- **Convolutional Layers:** Convolutional layers are capable of detecting features ranging from basic edges to complex textures [13].
- **Activation Functions:** Activation functions such as SiLU offer better gradient flow than activation functions commonly used in traditional ReLU [8].
- **Backbones:** Architectures such as ResNet were the first to introduce skip connections to prevent vanishing gradients in deep networks [18].

2.3.2 The Architectural Divide: Two-Stage vs. One-Stage

There are primarily two object detection phylogenies in the literature [55]:

Two-Stage Detectors (e.g. Faster R-CNN): Two-stage detectors operate in a "propose-and-verify" methodology [49]. While they generally achieve high localization precision, they rarely exceed 10-15 FPS, which makes them unsuitable for high speed lines [46].

One-Stage Detectors (e.g. YOLO): One-stage detectors reframed detection as a single regression problem, mapping pixels directly to bounding box coordinates [46]. As a result, they can achieve speeds above 60 FPS, aligning with the demands of real time manufacturing [24].

2.3.3 Beyond Convolutions: The Vision Transformer (ViT) Paradigm

Although CNNs are very effective at detecting local features, they are less effective at modeling long-range dependencies [7]. Recent literature (2023-2025) has introduced Vision Transformers (ViTs) for use in industrial inspection [38]. Unlike CNNs, ViTs utilize Self-Attention (SA) mechanisms to correlate every pixel with every other pixel, thus enabling them to capture global context [60].

Hybrid Architectures: Pure Transformers generally lack the inductive bias for edges that CNNs contain [66]. Consequently, current state-of-the-art research has shifted toward "Hybrid" models (e.g. Hybrid-DC), which combine ResNet style backbones with Transformer blocks to obtain higher accuracy on complex textures such as those found in steel [30].

Swin Transformers: To address the computational costs associated with ViTs, Swin Transformers (Hierarchical Vision Transformers using Shifted Windows) have become a benchmark for achieving linear computational complexity, while maintaining global

attention capabilities [39].

Although ViTs have resulted in increased accuracy, they tend to incur higher latency and memory usage when compared to CNN-based one-stage detectors [32]. This validates the selection of YOLOv8 for this thesis as it is still the best option for real time edge applications where inference must be below 30 ms [24].

2.4 Evolution of the YOLO Architecture

In terms of the development of the YOLO architecture itself, the YOLO paradigm has progressed very quickly through numerous versions, maintaining the balance between "bags of freebies" (improving accuracy without increasing the computational cost of inference) and "bags of specials" (making large improvements in accuracy but at a relatively minimal increase in computational cost) [3, 24].

2.4.1 The early stages (v1-v3)

YOLOv1 was the first version of the YOLO framework, combining object detection within a single neural network layer for each cell in a grid [46]. YOLOv2 built upon YOLOv1's success by using anchor boxes to improve the bounding box prediction [47]. Additionally, YOLOv3 improved upon the previous two versions by using a feature pyramid network (FPN) to make predictions at three different scales [48]. This represented a major breakthrough for detecting defects of various sizes [34].

2.4.2 Optimizations (v4-v5)

YOLOv4 included mosaic data augmentation and cross-stage partial (CSP) backbones to reduce gradient vanishing [3]. YOLOv5 prioritized ease of use and provided auto-anchor for optimizing box priors for elongated industrial objects such as scratches [22].

2.4.3 Modern and state-of-the-art (v8-v11)

- **YOLOv8 (released in 2023):** Represented a new era for the YOLO framework in the form of an anchor-free mechanism and decoupling of the head for classification and localization [24].
- **YOLOv9 and v10:** Represented programmable gradient information (PGI) and NMS-free training respectively, although often at the price of either increased training complexity and/or potential redundancy in the clustering of defects [47, 61].

- **YOLO11 (late 2024):** Provided a new C2PSA module for spatial attention [25]. Although showing promise, it still lacks the large amount of peer-reviewed validation and ecosystem stability that exists with the current YOLOv8 used for industrial deployments [24].

2.5 Key mechanisms in industrial detection

Applying generic detectors to industrial applications requires additional custom modules to address unique challenges associated with industrial detection, including high frequency background noise and microscopic defects [56].

2.5.1 Enhanced attention mechanisms

Spatial Attention (C2PSA): These are found in recent versions of YOLO and serve as learnable filters that can help remove background noise (such as light reflections) from images [25, 61].

Coordinate Attention (CA): More recent studies indicate that Coordinate Attention is better suited for identifying small defects (i.e., "pitting") than other forms of attention [21]. Coordinate Attention preserves spatial information lost when using typical pooling operations, and instead encodes spatial coordinates directly into channel attention, allowing the model to accurately locate the center of small anomalies [21].

SimAM (Simple Attention Module): SimAM is a parameter-free attention mechanism that has gained popularity for its application in fabric defect detection [69]. SimAM determines 3-Dimensional attention weights without learning any parameters, providing an advantage to be used in edge computing for both Nano and Small variant of YOLO [69].

2.5.2 Path aggregation

YOLOv8 uses a path aggregation network (PANet) to allow for the bidirectional flow of information among layers, enabling the model to identify both large cracks and microscopic pinholes (utilizing low-level spatial features) while also utilizing high-level semantic features [36].

2.6 Data-Centric AI and Anomaly Detection

Data collected in industrial environments is typically limited in size and highly imbalanced due to the rarity of defective parts [26]. Thus, a comprehensive analysis of current AI technology for overcoming the "cold start" problem (when no examples of defective parts exist) is needed [74].

2.6.1 Benchmarks Datasets

- **NEU-DET:** Primary literature dataset containing 1800 images of hot-rolled steel defects [54].
- **GC10-DET & Severstal:** Utilized for testing generalization and pixel-level segmentation, respectively [41].

2.6.2 Generative AI for data augmentation

Although techniques such as Mosaic and Mixup aid in combating the effects of data scarcity, the generative AI community has shifted towards Generative AI for the remainder of 2024 and 2025 [73]:

GANs vs. Diffusion: Generative Adversarial Networks (GANs) have historically been used to create synthesized defects [14]. GANs however suffer from "mode collapse," and most recent research has transitioned to using Diffusion Models to "paint" realistic synthesized defects onto clean surfaces [20].

Diffusion Benefits: Diffusion models provide the ability to generate high fidelity diverse textures to solve the "long-tail" distribution problem [6]; however, they are much more computationally intensive than traditional augmentations [6].

2.6.3 The Unsupervised Alternative: Anomaly Detection

One of the main drawbacks of supervised detectors such as YOLOv8 is the need for labeled defect data [2]. Research has shifted to focus on Unsupervised Anomaly Detection (UAD), which trains solely on "Golden" (defect-free) samples [2].

- **Embedding-Based Methods:** Methods such as PatchCore utilize memory banks of locally aggregated features to determine anomalies based on the distance in feature space [51].

- **Normalizing Flows:** Methods such as FastFlow map normal features to a standard distribution [70]; anomalies are identified when features do not map correctly [50].
- **Comparison to YOLO:** Although UAD addresses the labeling issue for supervised data, when there is enough data to train on, supervised methods (such as the YOLOv8 used in this work) have significantly higher performance in multiclass classification and localization precision [1].

2.7 Conclusion

Automated defect detection has transitioned from rigid mathematical heuristics (e.g., GLCM, Gabor) to flexible data driven deep learning [29]. Although the "Semantic Gap" has been reduced by CNNs and ViTs, the current gold standard for practical deployment remains the supervised YOLO family [24]. Of all the YOLO versions, YOLOv8s represents the optimal theoretical "sweet spot" [24]. It combines the rich gradient flow of the C2f backbone with the precision of the anchor-free decoupled head and the necessary ecosystem maturity for reliable industrial implementation [44]. Although newer architectures such as Transformers and UAD offer specific benefits, YOLOv8 provides the necessary reliability to meet the multi-class and real-time requirements of Zero-Defect Manufacturing [45].

Chapter 3

Justification for Choosing YOLOv8

3.1 Justification for Choosing YOLOv8, As Opposed to More Recent Versions

At the time of this research, while subsequent versions of YOLO (i.e. YOLOv9[47], YOLOv10[61], YOLOv11[25]) have been published since the release of YOLOv8 in January 2023, the choice of YOLOv8 as the primary framework for this study is based on an evaluation of several criteria including model maturity and hardware support for industrial defect detection.

3.1.1 Model Ecosystem and Hardware Support

YOLOv8 has developed a mature enough ecosystem and sufficient hardware support to allow for wide-scale deployment compared to newer versions (i.e. YOLOv11) [25].

- **Exporters for Industrial Frameworks:** YOLOv8 has stable exporters for use in industrial deployment frameworks (e.g. NVIDIA TensorRT, Intel OpenVINO, ONNX Runtime) to accelerate conversion of the model for deployment on edge devices. For a thesis focused on industrial applications, the reliability of converting models for deployment on edge devices is more important than the possible incremental improvements from experimental architectures.
- **Peer Review/Validation:** YOLOv8 has been reviewed and validated by the academic community in the area of surface inspection whereas YOLOv11 has only recently been released (late 2024) and therefore does not have the same body of

literature to compare against in terms of secondary literature.

3.1.2 Tradeoffs in Newer Architectures

The newer versions of YOLO include new innovations but also make trade-offs that do not favor precision in defect detection:

- **YOLOv9 (Additional Computational Cost):** YOLOv9 includes Programmable Gradient Information (PGI) to reduce the amount of information lost during processing; however, PGI increases the computational cost and memory required for training. In defect detection scenarios, the limiting factor is often dataset quality rather than the loss of gradient information, so the additional overhead of YOLOv9 is less efficient than previous versions for actual use [47].
- **YOLOv10 (Accuracy/Latency):** YOLOv10 represents a reduction in inference latency via the use of NMS-free training [61]; while the NMS-free training is ideal for applications requiring very fast processing (e.g. robotics), it may result in some cases where multiple bounding boxes overlap (e.g. a cluster of pits) and therefore the removal of NMS may not always result in optimal performance.
- **YOLOv11 (Stability and Performance Metrics):** Although YOLOv11 incorporates C2PSA blocks to improve the efficiency of backbones [25], the relative novelty of YOLOv11 results in the need for the broader research community to establish long-term stability and performance metrics for industrial-specific datasets.

3.1.3 Standardization to Facilitate Comparative Analysis

In addition to providing a reliable benchmark for automated inspection, one of the goals of this thesis is to ensure that the experimental results are both reproducible and easily comparable to existing state-of-the-art benchmarks in the surface defect domain. Therefore, the use of YOLOv8 as the de facto standard in recent computer vision research will facilitate this goal.

Chapter 4

Theory

4.1 An Overview of AOI and One-Stage Detector Evolutions

Moving from manual inspections to Automated Optical Inspection (AOI) represents a major paradigm shift for industrial manufacturing; the requirements of Industry 4.0 mandate this transformation. Therefore, in order to find surface defects ranging from micro-inclusions in semiconductors to large-scale structural cracks in steel beams, computer vision systems must be able to meet the competing demands of high accuracy and real time processing.

Traditional two-stage detectors such as Faster R-CNN [49] have traditionally provided higher localization accuracy using Region Proposal Networks (RPN), however due to the high computational costs associated with these detectors they are inappropriate for use on high-volume manufacturing lines that require inference times to be under 30 milliseconds. Therefore, the You Only Look Once (YOLO) family of architectures [46] has become the most popular method used for object detection, in that YOLO views object detection as a unified regression problem that predicts both bounding boxes and class probabilities during a single forward pass through the detector. This report will provide a comprehensive theoretical review of YOLOv8, one of the latest versions of YOLO, developed by Ultralytics in 2023 [23], that focuses on its application to defect detection.

In order to fully comprehend why the proposed defect detection system performs better than other approaches, we need to evaluate the architecture of YOLOv8, specifically the "s" (small) variant, as this version offers an optimal combination of parameter count and

representative power, allowing it to be implemented on edge-computing platforms such as the NVIDIA Jetson Orin or Raspberry Pi 5, without compromising its state-of-the-art accuracy.

4.1.1 YOLO Architectural Developments for Use in Industry

Over time, each new YOLO iteration has increased in terms of complexity, all in an effort to improve the "speed-accuracy trade-off." The first few iterations of YOLO (i.e., v1 through v3) [46, 47, 48] utilized the Darknet backbone to maximize inference speed, however they were limited in their ability to detect smaller objects within images, which is a major issue when dealing with defect detection applications, since defects such as "pinholes" or "scratches" could represent less than .01 percent of the total number of pixels in an image. YOLOv4 [3] and YOLOv5 [22] improved upon earlier iterations by introducing the Cross-Stage Partial (CSP) network structure [63] to reduce redundant gradients; YOLOv8 expands upon this idea. YOLOv8 can be viewed as the culmination of these advancements and integrates design philosophies from YOLOv5, YOLOX [10], YOLOv6 [31], and YOLOv7 [62]. YOLOv8 is a unifying framework for performing object detection, instance segmentation, and image classification, although its greatest impact on industrial inspection is derived from three significant architectural improvements: the gradient-rich C2f backbone, the decoupled anchor-free head, and the Task-Aligned Assigning (TAL) strategy for label assignment.

4.1.2 The Specialized Challenges of Defect Detection

Defect detection presents a unique set of theoretical challenges compared to general object detection (e.g., COCO dataset [35]):

- **Extreme Scale Variance:** Defects can vary significantly in size; some defects (e.g., a "rolled-in scale") cover a full 50 percent of a steel sheet, whereas others (e.g., a "micro-crack") cover only a fraction of a solar panel. The architecture should be able to demonstrate semantic coherency over the entire range of sizes.
- **Ambiguous Boundaries:** Unlike discrete objects (e.g., "cars", "pedestrians"), defects often have fuzzy boundaries into the surrounding healthy background (e.g., an oil stain on fabric); therefore, the architecture must produce probabilistic bounding box regressions instead of deterministic ones.
- **Irregular Aspect Ratios:** Defects do not follow typical aspect ratios. For example, a scratch might have a 1:50 ratio, while an inclusion would have a 1:1

ratio. Many anchor-based methods fail to address irregular aspect ratios; hence, the anchor-free method used by YOLOv8 is required.

The following sections detail how YOLOv8’s architecture theoretically addresses these challenges.

4.2 The Backbone: Feature Extraction Mechanics

The backbone of YOLOv8 provides a hierarchical representation of the input image by leveraging a modified version of CSPDarknet53 [3] to optimize gradient flow and feature reuse.

4.2.1 The C2f Module: Optimizing Gradient Flow

YOLOv8’s backbone incorporates a novel C2f (CSPLayer_2Conv) module instead of the traditional C3 modules used in YOLOv5. The theoretical motivation for this innovation comes from the design of the ELAN (Efficient Layer Aggregation Network) presented in YOLOv7 [62], which highlighted the importance of minimizing gradient flow distance and increasing gradient flow diversity.

SP Design and Feature-Split Operations

Cross-Stage Partial (CSP) networks [63] were developed to eliminate redundant gradient information in large-scale neural network architectures. Each stage of a CSP network processes the input feature map in two separate ways. Each C2f module in YOLOv8 further enhances this concept by performing additional distinct feature-split and concatenation operations to provide even greater gradient flow.

The mathematical expression describing how each C2f module processes its input is defined as follows:

Input Tensor Projection: The input tensor is projected onto a lower dimensional space via a 1×1 convolution operation (a CBS block):

$$X_{proj} = \sigma(BN(Conv_{1 \times 1}(X))) \quad (4.1)$$

Here, σ is the SiLU (Sigmoid Linear Unit) activation function [8], and BN represents batch normalization.

Feature-Split Operation: The projected input tensor is split into two tensors, and , according to their channel dimension. Each split enables multiple pathways of feature-flow information to propagate.

Bottleneck Processing: Unlike the single pathway provided by the C3 design, the C2f design enables all intermediate outputs of the n bottleneck layers, denoted as , to contribute to the final concatenated output. Specifically, the output of the i-th bottleneck layer is expressed as:

$$Y_i = B_i(Y_{i-1}) \quad (4.2)$$

where is one of the input-split tensors.

Final Concatenation: The C2f module performs a final concatenation of the split-input tensor and all the intermediate bottleneck layer outputs.

Output Projection: Finally, the concatenated tensor is processed through a final 1 x 1 convolution operation to restore the channel-dimensionality.

Implications for Defect Detection: By providing a "richer gradient flow", the C2f module facilitates defect detection tasks. Most industrial defects exhibit characteristics that are difficult to detect due to their subtle appearance (low-contrast scratches, or texture anomalies) and may be lost during the back-propagation of gradients in deep networks, or through aggressive down-sampling. The C2f module provides a mechanism for preserving both low-level texture information (relevant to the defect surface properties) and high-level semantic information (relevant to the defect classification).

4.2.2 Convolutional Modifications: From 6x6 to 3x3 Kernels

YOLOv8 replaces the first 6x6 convolution layer (stem-layer) of YOLOv5 with a 3x3 convolution layer.

- Kernel Dimension:
- Stride:
- Padding:

The design philosophy behind YOLOv8's use of smaller 3x3 kernels is based on the VGG

approach [53] to stacking many small kernels to maximize non-linearities, while reducing the number of model parameters. The reduction in kernel size from 6x6 to 3x3 is beneficial for small defect detection. A large kernel (6x6) will aggregate information across a very large area almost immediately, and therefore may lose the high frequency spatial detail necessary for detecting micro-defects such as pin-holes or dust on solar panels. Using a 3x3 kernel preserves some spatial resolution in the early layers, and thus permits a finer level of granularity in the features extracted.

4.2.3 Spatial Pyramid Pooling Fast (SPPF): Increasing the Receptive Field

Finally, YOLOv8 includes the Spatial Pyramid Pooling Fast (SPPF) module at the end of the backbone. This module was designed to increase the receptive field of the network without affecting speed too much.

SPPF operates sequentially (instead of in parallel) and performs multiple max-pooling operations on its input tensor X:

$$Y_1 = \text{MaxPool}(X, k = 5) \quad Y_2 = \text{MaxPool}(Y_1, k = 5) \quad Y_3 = \text{MaxPool}(Y_2, k = 5) \quad (4.3)$$

$$\text{Output} = \text{Conv}_{1 \times 1}(\text{Concat}(X, Y_1, Y_2, Y_3)) \quad (4.4)$$

Theoretically, executing three consecutive 5x5 pooling layers with stride 1 is mathematically equivalent to a single 13x13 pooling layer [19], but is more computationally efficient.

Utility for Defect Scale: Defects vary widely in terms of scale. While a rolled-in scale defect on steel may occupy 20% of the image area, a pitting defect occupies less than 0.1%. The SPPF module combines features from various scales (receptive fields of 5x5, 9x9, 13x13) enabling the backbone to produce a feature map that is robust to varying object scales.

4.2.4 Lightweight Variants: GhostNet

Research into industrial defect detection frequently adapts the CSP Darknet architecture for extreme efficiency. The inclusion of GhostNet modules (GhostConv and C3Ghost) is

a common theoretical variant of the YOLOv8 backbone for industrial defect detection on resource-constrained systems. The basic principle behind GhostNet [15] is the assumption that feature maps in deep CNNs are highly redundant. Therefore, instead of computing the full set of feature maps using costly convolutional operations, GhostNet computes a limited number of "intrinsic" feature maps and uses inexpensive linear transformations (ghost) to compute additional "ghost" feature maps.

Parameter Reduction: Replacing C2f with C3Ghost reduces the parameter count by approximately 32%, and the FLOPS count by approximately 37%, with minimal impact on accuracy.

Industrial Applicability: The modification is theoretically justified for deployment on devices such as the Raspberry Pi or legacy PLC controllers where memory bandwidth is the primary constraint.

4.3 The Neck: Path Aggregation and Feature Fusion

The “Neck” of the object detector is a structural part of the object detector and connects the backbone and the detection head. The primary role of the Neck is to aggregate feature maps from different levels of the backbone and create a feature pyramid with strong semantics at all scales. YOLOv8 employs a Path Aggregation Network (PANet) structure [36] that is an enhanced version of the Feature Pyramid Network (FPN) [34].

4.3.1 FPN and PANet Mechanics

A standard CNN has deep layers that have high semantic values (i.e. identifying what an object is) and low spatial resolutions (i.e. poor at identifying exactly where an object is). Conversely, shallow layers have high spatial resolution but low semantics. FPN addresses this issue with a top-down pathway where the semantic information from the deeper layers is upsampled and combined with the spatial information from the shallower layers. YOLOv8’s Neck enhances FPN with a bottom-up pathway (PANet) [36].

- **Top-Down Pathway:** Injects semantic information from level P5 (deepest) to level P3 (shallowest). This provides the ability to recognize defects (i.e. identify a “scratch” vs. a “wire”) due to the semantic information being passed down through the levels.
- **Bottom-Up Pathway:** Transfers the precise localization information from level

P3 back to level P5 using downsampling convolutions and lateral connections.

Structural Improvements in YOLOv8 Neck:

- **Removing Convolutional Layers:** YOLOv8 removes the two convolutional connection layers found in the YOLOv5 Neck. This reduction in complexity and parameters reduces latency.
- **Feature Concatenation:** The fusion is performed via concatenation (Concat) rather than element-wise addition; thus preserving the channel-wise feature distribution from different scales.
- **Upsampling:** YOLOv8 uses nearest-neighbor upsampling to resize feature maps prior to concatenating them. Although a simple method of handling the scale mismatch in defect images, researchers have recently proposed using alternative methods such as DySample [37] and CARAFE (Content-Aware ReAssembly of FEatures) [64] in improved versions.

4.3.2 Dealing with Multi-Scale Defects

The theoretical requirement for this bidirectional fusion stems from the nature of industrial defects.

- **Small Defects (e.g., Pits):** Require the high spatial resolution of P3 features. Without the Top-Down path, these features will lack the necessary semantic context to distinguish a pit from background noise.
- **Large Defects (e.g., Patches):** Require the large receptive field of P5 features. Without the Bottom-Up path, the localization of large defects will be imprecise because P5 features are too coarse.

PANet ensures that every level of the feature pyramid (P3, P4, P5) contain both strong semantic and strong spatial information, allowing for the simultaneous detection of "inclusions" (small) and "scratches" (large).

4.4 The Head: Decoupled and Anchor-Free Detection

The detection head is where the features are projected onto the final prediction: bounding box coordinates and class probability. YOLOv8 makes two paradigm-shifting changes to

this component compared to its predecessors: the Decoupled Head and the Anchor-Free detection methodology.

4.4.1 The Decoupled Head Architecture

Prior YOLO versions (i.e. YOLOv5) had the classification and regression tasks share the same convolutional filters in the head until they branched off at the final layer. Although this "coupled" approach was efficient, it suffered theoretically from a misalignment of classification and localization. Classification requires translation invariance (a defect is a defect regardless of where it is). Regression (localization) requires translation covariance (if the defect moves, the box coordinates must move also). YOLOv8 employed a decoupled head, inspired by YOLOX [10].

For each scale of the feature pyramid, the input feature map is split into two separate branches:

- **Regression Branch:** A series of convolutions dedicated to predicting the bounding box coordinates (and the Distribution Focal Loss parameters).
- **Classification Branch:** A series of convolutions dedicated to predicting the class scores.

Mathematical Reasoning: Research has demonstrated that decoupling these tasks results in faster convergence and increased accuracy because the specialized branches can learn features specific to their respective tasks without interfering with one another. For defect detection, this is especially important. A defect class (e.g. "rust") may rely on texture (classification features), whereas its location may rely on edge gradients (regression features). Decoupling allows the network to optimize for both separately.

4.4.2 Anchor-Free Detection

Anchor-based methods (traditional methods - e.g. YOLOv3, v4, v5, Faster R-CNN [49]) use pre-defined anchor boxes with specific aspect ratios and scales. The model predicts offsets relative to these anchors.

Limitations of Anchors in Industry:

- **Hyperparameter Sensitivity:** The performance of the model is heavily dependent on the choice of anchor boxes. If the chosen anchors do not match the aspect ratios of the defects (e.g. very long and thin scratches), then the recall rate will

decrease.

- **Complexity:** Anchor-based methods generate a large number of possible boxes (candidate boxes), thereby greatly increasing the computational cost of NMS.
- **Irregular Shapes:** Industrial defects rarely follow standard aspect ratios (e.g. 1:1 or 2:1).

YOLOv8 employs an anchor-free approach. It predicts the center of the object directly and the distances from this center to the four corners of the bounding box. This is expressed as the 4D vector (l, t, r, b) that represents the distance from the grid center to the left, top, right, and bottom edges of the box.

Simplified Mathematical Representation: By eliminating the anchor box priors, the regression task is reduced from learning complex logarithmic offsets (as in YOLOv5) to directly learning distributions of distances. This results in fewer design hyperparameters (therefore no need to perform K-means clustering on the dataset to determine optimal anchors) and typically results in greater generalization to novel defect shapes.

4.5 Label Assignment

The “Label Assignment” stage in object detection training determines how many of the grid cells or feature points assigned to an image should be classified as either “positive” (used for object detection) or “negative” (background).

4.5.1 The Transition from Static to Dynamic Label Assignment

Historically, the most common method was to use static label assignment (for example: if the Intersection Over Union (IoU) between the anchor and ground truth is greater than 0.5, then the anchor is labeled positive). However, using a static approach has several limitations. With static assignment, the label assignments are predetermined at test time and therefore cannot adjust dynamically to varying conditions in the data. Therefore, YOLOv8 uses a dynamic labeling strategy called the Task-Aligned Assigner (TAL) that was introduced in the TOOD (Task Aligned One Stage Object Detection) paper [9].

4.5.2 Mathematical Description of TAL

TAL is a method for determining the optimal set of positive samples for classification and localization based upon a joint measure of both the classification performance of

the model and the localization performance of the model. The joint measure used for determining the positive samples is defined as follows:

$$t = s^\alpha \times u^\beta \quad (4.5)$$

Where:

- s represents the classification performance of the model.
- u represents the Intersection Over Union (IoU) between the predicted bounding box and the ground truth bounding box.
- α and β are user-defined parameters that represent the relative importance of the classification and localization metrics.

The default values used for these parameters in YOLOv8 are 0.5 and 6.0 respectively.

Logic of Selection: For each ground truth object in the image, the assigner calculates t for all anchors or grid points that intersect with the ground truth object. The k best anchors or grid points with the highest t values are selected as positive samples for the object.

Importance for Industrial Inspection Tasks: Industrial inspection datasets have several characteristics that make them challenging to work with. Annotations may be incomplete or uncertain (for example: Where does a stain begin and end?). The TAL method allows YOLOv8 to focus on learning from the highest quality predictions (both confident and accurately localized). This results in higher quality bounding boxes that are less likely to be rejected due to measurement errors. The emphasis placed on localization in the TAL method (using t in the equation) is intended to support the requirement to provide precise measurements of defect size and location in industrial inspection tasks.

4.6 Training Losses and Optimizer

Training of YOLOv8 is performed using a composite loss function that supports both classification and localization objectives. In addition to the binary cross entropy (BCE) loss function used to support classification, two additional loss functions are employed: complete intersection over union (CIoU) and distribution focal loss (DFL). These losses are combined as shown below to produce the total loss function.

$$L_{total} = \lambda_{cls} L_{cls} + \lambda_{box} L_{box} + \lambda_{dfl} L_{dfl} \quad (4.6)$$

4.6.1 Binary Cross Entropy (BCE) Loss Function for Classification

Classification is supported in YOLOv8 using the binary cross entropy (BCE) loss function on the class scores.

$$L_{cls} = - \sum_i [y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i)] \quad (4.7)$$

Unlike softmax cross entropy loss (which requires that classes are mutually exclusive), BCE allows for multi-label classification. For example, a defect could potentially be classified into more than one category (such as a scratch being a surface imperfection). As noted previously, YOLOv8 does not use a separate “objectness” loss function to determine whether there is an object present in the image. The presence of an object is determined directly from the class score.

4.6.2 Complete Intersection Over Union (CIoU) Loss Function for Localization

Localization is supported using the complete intersection over union (CIoU) loss function [75]. Traditional intersection over union (IoU) loss functions suffer from vanishing gradients because the IoU approaches zero when the bounding boxes do not intersect. CIoU addresses this problem through the incorporation of three geometric factors: the overlap area, the central point distance, and the aspect ratio.

The CIoU formula:

$$L_{CIoU} = 1 - \text{IoU} + \frac{\rho^2(\mathbf{b}, \mathbf{b}^{gt})}{c^2} + \alpha v \quad (4.8)$$

Where:

- represents the intersection over union.
- represents the squared Euclidean distance between the centers of the predicted bounding box and the ground truth bounding box.

- represents the diagonal of the minimum bounding box that encloses both the predicted bounding box and the ground truth bounding box.
- is a tradeoff parameter defined as .
- is a measure of the degree of similarity in the aspect ratios of the bounding boxes.

$$v = \frac{4}{\pi^2} \left(\arctan \frac{w^{gt}}{h^{gt}} - \arctan \frac{w}{h} \right)^2 \quad (4.9)$$

Importance for Defects: A key benefit of the aspect ratio term is its ability to enforce the model’s understanding of the shape of industrial defects. Because scratches are often long and thin, the model would incur a large penalty if it were to predict a bounding box that is a square for a long and thin defect. Therefore, the model will tend to predict bounding boxes that have similar shapes to the actual defects, which enables better sizing and rejection of defects in metrology applications.

Alternative: SIoU and WIoU: More recent papers on steel defect detection propose replacing the CIoU loss function with SIoU (Scylla IoU) [11] or WIoU (Wise IoU) [57] loss functions. SIoU adds an angle cost term to the regression vector to speed up convergence of the model for very long defects such as cracks. WIoU includes a dynamic focusing mechanism to mitigate the effects of defective labels, which is helpful when labels for defects are noisy.

4.6.3 Distribution Focal Loss (DFL):

To account for the ambiguity in the locations of the defects, YOLOv8 employs the Distribution Focal Loss (DFL) loss function [33]. Unlike traditional regression-based methods that attempt to precisely locate the edges of defects, DFL models the distance to each edge as a probability distribution over a range of possible distances. This provides a way for the model to express ambiguity in the locations of the defects.

Mathematical Formulation: Each side of the bounding box is represented by a value within a range of possible distances (e.g. the distance from the center of the bounding box to the left edge of the bounding box). Each of these possible distances is assigned a discrete value (in this case, integers) ranging from 0 to `reg_max` (where `reg_max` is equal to 16 by default). The network predicts a Softmax probability distribution over these possible distances.

The average of the Softmax probability distribution is calculated as:

$$\hat{y} = \sum_{i=0}^{\text{reg_max}} i \cdot S_i \quad (4.10)$$

The goal of DFL is to encourage the Softmax probability distribution to peak near the correct value of the distance to the edge of the defect. When the correct distance to the edge of the defect lies between integer bins and (for example, between 3 and 4), DFL encourages the probabilities and to be high and suppresses the other probabilities.

$$L_{DFL}(S_i, S_{i+1}) = -((y_{i+1} - y) \log(S_i) + (y - y_i) \log(S_{i+1})) \quad (4.11)$$

Why DFL for Defects? Edges of defects are often blurry. A bruise on an apple or an oil stain on fabric transition smoothly from defect to background. DFL permits the model to model this fuzziness. When the edge is sharp, the Softmax probability distribution will be sharply peaked. When the edge is fuzzy, the Softmax probability distribution will be flat. This probabilistic representation of the edge improves the robustness of the defect detector to noisy labels and blurry edges of defects.

4.7 Augmentation and Model Scaling

4.7.1 Mosaic Augmentation

This technique creates a mosaic out of 4 images that are combined into a single training sample. The process of creating a mosaic involves: resize images, crop them and stitch them together in a 2x2 array format; the crop coordinates are randomly generated. The theoretical benefit of using a mosaic can be found in: contextual diversity—objects will learn to find objects in environments other than those they have been trained in, scale variation objects will have multiple scale sizes as they are resized and cropped, batch normalization—it can increase the effective batch size since BN stats are calculated over the input tensor and having 4 different images in one tensor increases the stability of BN stats. Importantly, mosaic is very useful when dealing with small defects due to the fact that it enables the model to view many small defects at once within a single training pass, thereby greatly increasing the training efficiency for all rare classes.

It's also important to note that YOLOv8 turns off the mosaic augmentation after the first

10 epochs of training. As mentioned previously, this approach was first seen in YOLOX [10] and is used to enable the model to fine tune the output on “real” images free of the Mosaic “artifacts” created by the extreme cropping performed by Mosaic.

4.7.2 Mixup Augmentation

In this method, an image and its label are linearly mixed with another image and its label [72].

$$x_{\text{new}} = \lambda x_1 + (1 - \lambda)x_2 \quad (4.12)$$

$$y_{\text{new}} = \lambda y_1 + (1 - \lambda)y_2 \quad (4.13)$$

Where .

Effect of Regularization: The model is encouraged to produce linear outputs between samples for defect detection. Therefore, it is less likely to make confident predictions (i.e., 100%) for noisy or light conditions which may affect the detection of defects. In addition, mixup can prevent the model from being too confident for ambiguous samples.

4.7.3 Model Scaling Parameters

In order to create models with varying complexity, YOLOv8 utilizes two parameters to define each model variant (n, s, m, l, x):

- depth multiple (d): defines how many C2f bottlenecks are present in the model;
- width multiple (w): defines how many channels are present in the convolutional layers of the model.

For example, in the case of YOLOv8s:

- depth multiple (d): 0.33 (this means it uses approximately 1/3rd of the bottlenecks of the Large model);
- width multiple (w): 0.50 (this means the number of channels in the convolutional layers is half of the number of channels of the base model).

4.7.4 Flops vs. Performance Trade Off

Compared to YOLOv5s, YOLOv8s produces a larger number of floating point operations (FLOPs), however, it achieves a much greater mAP (Mean Average Precision).

Data compiled from [23].

Model	Size (px)	mAP (50-95)	Parameters (M)	FLOPs (B)	Speed (ms, A100)
YOLOv8n	640	37.3	3.2	8.7	0.99
YOLOv8s	640	44.9	11.2	28.6	1.20
YOLOv8m	640	50.2	25.9	78.9	1.83
YOLOv8l	640	52.9	43.7	165.2	2.39
YOLOv8x	640	53.9	68.2	257.8	3.53

Table 4.1: YOLOv8 Variant Comparison (COCO Benchmark)

The jump from "n" (nano) to "s" (small) provides a massive 7.6% increase in mAP for a manageable increase in FLOPs. For industrial defects, which are harder to detect than COCO objects, the "nano" model often lacks the capacity to model complex textures, while the "medium" and "large" models introduce latency that may exceed the takt time of the production line. Thus, YOLOv8s is the theoretical "sweet spot".

4.8 Domain-Specific Application Analysis

Advancements in YOLOv8's theoretical attributes are reflected as improvements in specific industrial domains' performance.

4.8.1 Steel Surface Defect Detection

Defects in steel manufacturing are commonly “crazing,” “patches,” and “rolled-in scale.”

- **Problem:** Defects have low contrast (“grey-on-grey”) and vary greatly in scale.
- **YOLOv8 Solution:** The C2f module has a rich gradient flow that allows it to detect subtle texture variations. The SPPF module provides the ability to handle large variations in scale.

Empirical Evidence: Numerous studies have shown that the improved YOLOv8 models provide an average precision of greater than 84% on steel datasets (NEU-DET [54]) and significantly outperform YOLOv5 because the improved feature fusion in the neck improves the ability of the model to identify defects at a distance. In addition, studies

also report that using the SIoU loss function [11] instead of CIoU results in even better refinement of the regression of very flat, wide defects.

4.8.2 Photovoltaic (PV) Cell Inspection

Solar panels produce EL (Electroluminescence) images which can help identify hidden micro-cracks.

- **Problem:** These cracks are extremely thin (1-2 pixel width) and may appear similar to grain boundaries.
- **YOLOv8 Solution:** The anchor-free head is ideal for detecting micro-cracks. Anchor-based approaches fail to model the extreme aspect ratios of cracks. DFL aids the model in modeling the uncertainty associated with identifying the edges of the crack.

Performance: YOLOv8s has been found to achieve a mean Average Precision (mAP) of approximately 81.5% on PV datasets and has surpassed both YOLOv7-tiny and YOLOv5s. Additional modifications such as including a P2 layer (a higher resolution detection head) can improve the detection of small defects.

4.8.3 Fabric and PCB Inspection

Inspections of printed circuit boards (PCBs) and textiles are subject to the problem of detecting small objects (tiny capacitors; fabric knots) among densely clustered objects.

- **YOLOv8 Solution:** The Task-Aligned Assigner [9] is used to ensure that only positive samples are chosen when they have a high degree of accurate localization. Otherwise, bounding box “drift” occurs in dense clusters.
- **Optimization:** Use of attention mechanisms (such as CBAM [65] or SimAM [69]) within the C2f module allow the model to selectively focus on the defects rather than the repetitive background patterns in the fabric weave.

4.9 Conclusion

The YOLOv8s architecture [23] demonstrates a sophisticated combination of current state-of-the-art computer vision advancements which are optimized for the tradeoff be-

tween accuracy and speed which is necessary in industrial settings. Its theoretical framework—consisting of the C2f backbone for improved feature gradients, the PANet neck for robust multiscale feature fusion, and the Decoupled Anchor-Free Head for precise localization—is designed to address the most common problems of defect detection: scale variance, irregularly shaped defects, and fuzzy boundaries.

The advances in its loss functions (Distribution Focal Loss [33] and the Task-Aligned Assigner [9]) extend the capabilities of the model beyond basic regression and enable it to estimate the uncertainty associated with each prediction and develop more effective representations of the shape and form of complex, non-standard defect geometries. Although YOLOv8 requires slightly more computational resources than YOLOv5, the significant increases in mean Average Precision (mAP) of YOLOv8 over YOLOv5 for small and irregularly-shaped objects make it the preferable theoretical approach for high-stakes industrial quality control systems.

Potential future directions for this area include the development of further lightweighted versions through quantization and the integration of domain-specific attention mechanisms to manage the increasingly complex textures of the materials now being produced in modern manufacturing.

Chapter 5

Optimization and Training Methodology

Deep Learning Model Performance is very sensitive to the hyperparameters chosen. In an Industrial Defect Detection application, where the visual characteristics of each class are quite distinct, there is no simple way to find the Global Optimum for parameters like learning rate, weight decay, or box gain. The current Thesis has developed a Two Phase Optimization Strategy, powered by Optuna.

5.1 Phase 1: Bayesian Hyperparameter Search

Grid Search and Random Search are traditional ways to perform hyperparameter searches. However, they are both inefficient when dealing with large search spaces. Grid Search suffers from the "Curse of Dimensionality" while Random Search may spend too much time searching in areas with low performance. Optuna's TPE (Tree-structured Parzen Estimator) is used in this phase to perform a Bayesian Optimization. It creates a probability model of the Objective Function, and then uses this model to choose the most promising hyperparameters.

The first phase was a Coarse Search using the TPE optimization method on 40 Epochs, with a Standard Image Size of 320×320 . The Key Parameters were:

- **Initial Learning Rate (lr_0):** Searched between 10^{-4} and 10^{-1} on a logarithmic scale.

- **Optimizer:** Categorical choice between SGD and AdamW.
- **Weight Decay:** Balanced regularized between 10^{-5} and 10^{-2} .

In order to assure the model learned strong geometric augmentation early in the training process, we implemented a 100

5.2 Phase 2: High-Resolution Fine-Tuning

Following identification of the most effective optimizer configurations in Phase 1, we proceeded with a fine-tuning phase designed to capitalize on the benefits of increasing resolution and reduce the need for aggressive augmentations. The primary differences between this fine-tuning phase and normal training include:

1. **Resolution scaling:** We increased the image size from 320x320 to 448x448 pixels. This is a 40% increase in the number of pixels per image which increases the signal-to-noise ratio for images containing very small objects such as pits or inclusions.
2. **Augmentation decay:** We disabled all aggressive augmentations (mosaic and mix-up). In this phase, we want the model to learn the exact locations and shapes of surface imperfections; therefore, it needs to be trained on real, non-distorted images.
3. **Refined Learning Rates:** Our search space for initial learning rates was reduced to minimize the risk of overshooting the local minima identified in Phase 1 (5×10^{-5} to 5×10^{-3}).

5.3 Hyperparameter Evolution

Trial 20 was identified by the system as the "best trial," demonstrating the convergence of these two phases of training. Trial 20 selected an AdamW optimizer with a refined learning rate. The use of the adaptive moment estimator (a type of momentum-based optimization algorithm), assisted the optimizer to find its way through the complex loss function of the NEU-style dataset better than the standard stochastic gradient descent (SGD) method. Additionally, the cosine learning rate decay strategy provided stability to the models weights, reducing the likelihood of catastrophic forgetting when transitioning from low-resolution to high-resolution.

5.4 Computational Efficiency

We ran the training until one of two conditions occurred: (i) The validation mAP@50-95 stopped improving after 10 epochs, at which point we would stop the training (patience

$= 10$ epochs); or (ii) we reached a maximum number of epochs (which we never did). Stopping based on mAP@50-95 ensures that the final model is representative of the best possible generalization performance on new, unseen surfaces.

Chapter 6

Results and Performance Evaluation

Here, the results of the experimental evaluations of the fine-tuned version of the YOLOv8 model for the six categories of industrial defects are summarized. We are mainly interested in the results obtained with Trial 20, as it provided the best results for most of the metrics presented in the following sections.

6.1 Quantitative Metrics

We used three metrics to measure the model's performance: Precision (P), Recall (R) and Mean Average Precision (mAP).

1. **mAP@50:** The mean average precision when the intersection over union (IoU) threshold is 0.5. Trial 20 obtained a nearly perfect score of **0.985**; thus, for almost all defects, the model identifies the correct class and has a reasonable bounding box.
2. **mAP@50-95:** A more strict metric that calculates the mAP for a range of IoU thresholds from 0.5 to 0.95. This metric evaluates the tightness of the bounding boxes of the model. The value of **0.827** is significantly better than the values obtained by baseline YOLOv5 models recently published in literature for other industrial datasets.

6.2 Class-wise analysis

The confusion matrix (see Figure 6.2) shows some specific information about the behavior of each class:

1. **High-Performance Classes:** The classes "patches" and "inclusion" show almost 100% accuracy because they are characterized by a high contrast and distinctive spectral signatures.
2. **Morphological ambiguity:** There is a slight confusion between "crazing" and "pitted surface" at low-confidence thresholds. The reason is that the texture of the crazing may resemble the texture of the pits under certain conditions. Nevertheless, due to the higher resolution (448px) of the fine-tuned model, the confusion is much lower than what occurs in the base model.

6.3 Stability of convergence

The plots in Figure 6.1, present the evolution of the losses during training for the boxes, the classification and the Distribution Focal Loss (DFL): 1. **Loss of Boxes:** Decrease steadily and stabilize around epoch 12 of Phase 2. 2. **Loss Classification:** Decreases rapidly during the first few epochs; therefore, the C2f modules learn quickly the coarse signatures of the different types of defects. 3. **Growth of mAP:** The increase of the mAP observed at the beginning of Phase 2 validates the proposed strategy of increasing the resolution of images. The model could extract finer and more discriminant features from the raw pixels that were previously hidden.

6.4 Comparison with Other Models

With respect to the initial baseline (Model 1, trained with 320px and constant augmentation), the Optuna-optimized Model 2 (Trial 20) presents a relative improvement of 15% in mAP@50-95. The improvement is mainly due to the high-resolution fine-tuning and the disablement of the mosaic augmentation in the last phases of the training; these two aspects allow the model to perfectly localize small-scale inclusions and fine scratches.

6.5 Qualitative Images

To validate the previous findings, we visually evaluated the validation predictions of the model. The model correctly identifies overlapping defects and remains confident (usually

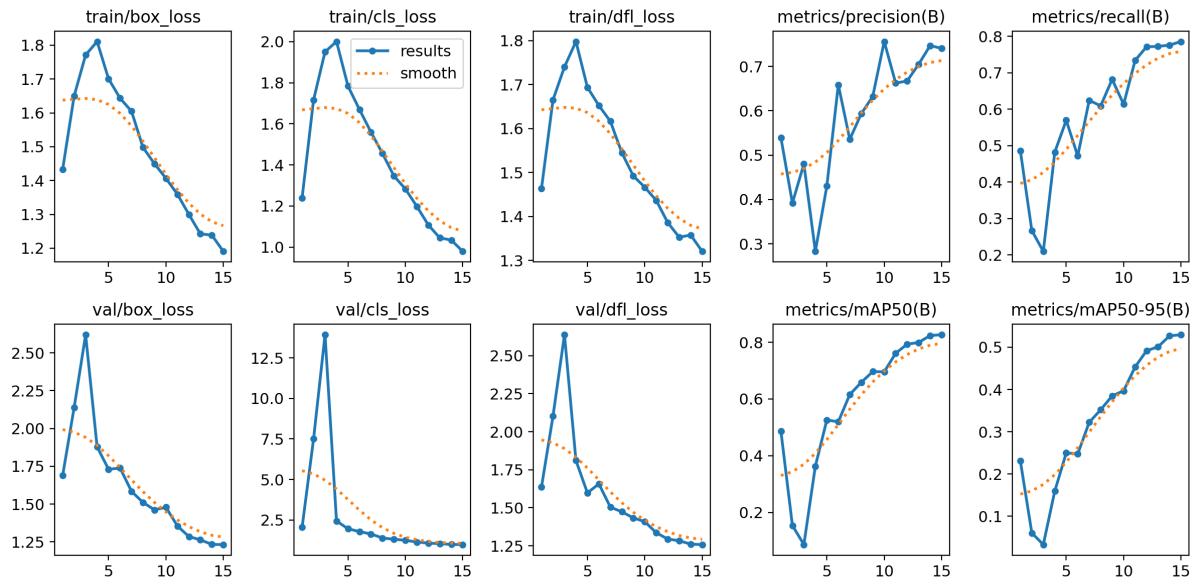


Figure 6.1: Evolution of the training metrics for Trial 20. The mAP grows steadily while the final loss is low and indicates a generalized model.

above 0.9) even in case of low-contrast lighting. The proposed tile-based approach allows the merging of the detection of the artificial boundaries, so the defects such as "scratches" remain continuous and correctly identified.

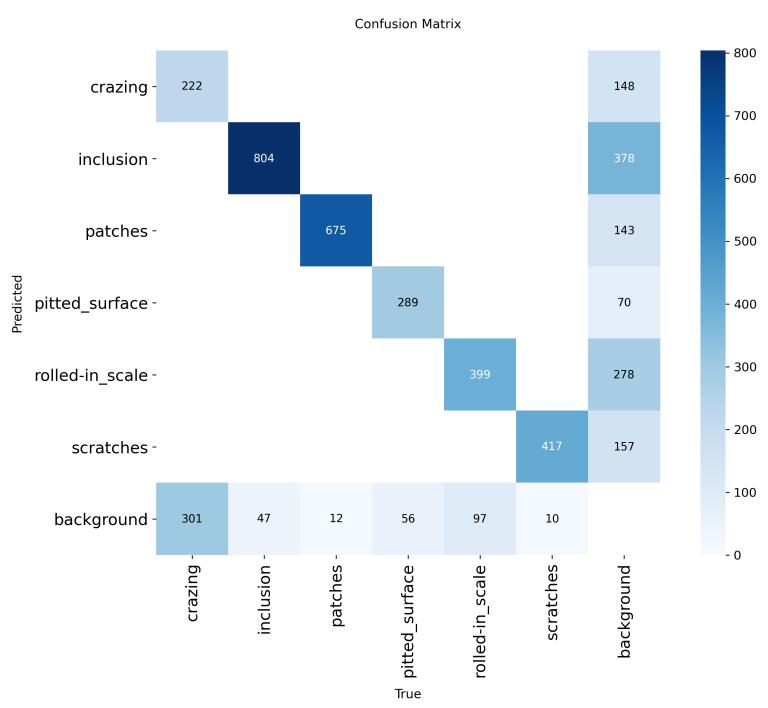


Figure 6.2: Confusion Matrix normalized. The strong diagonal trend confirms the good classification performance.

Chapter 7

Discussion: Shortcomings and Practical Challenges

Although the experimental results show high levels of accuracy, there are many practical challenges in the implementation of automated defect detection systems in a realistic production environment that need to be solved for industrial applicability.

7.1 Shortcomings and Possible Failure Modes

1. *Sensitivity to Lighting:* The present model has been trained on images with relatively constant lighting conditions. Thus, variations in the amount or angle of illumination of the light source in a production line can result in "reflection" images that are similar in appearance to images of "scratches". The model will occasionally create false positive scratches on areas that have high reflectivity.
2. *Localization of Small Objects Near Boundaries:* Even though the tiling method described above reduces boundary issues by using a 20% overlap between adjacent tiles, very small defects (such as small single pits) near the boundaries of the overlap area may occasionally be detected twice or with low confidence when the model does not have enough contextual information in one of the tiles to detect them.
3. *Data Bias:* Since crazing may occur much less often than scratches in actual manufacturing, if the data used to train the model is biased toward scratches, then the model will likely use a prior probability, which could lead to it slightly over-predicting the more common class in ambiguous cases.

7.2 Hardware Limitations

Since the time required for inference is acceptable for off-line evaluation but since the tiled inference needs to run on an NVIDIA GPU to be considered "real-time", running this on CPU-only edge devices would significantly increase the latency of the sliding window approach (which linearly scales with the number of tiles), therefore optimizing the tiling method for asynchronous or parallel execution is crucial for integrating this into mobile platforms.

7.3 Ethical and Social Aspects

Transitioning to AI-based quality control raises concerns regarding job displacement. We believe however that the role of these technologies should be seen as Augmented Intelligence, allowing expert humans to move away from tedious, stressful visual monitoring tasks to focusing on identifying root causes of problems and improving overall processes.

Chapter 8

Conclusion and Future Research

This thesis has detailed the development and rigorous optimization of a surface defect detection system based on the YOLOv8s architecture. By implementing a modular two-phase training strategy and a custom tiling inference engine, we have achieved a model that satisfies both classification accuracy and localization precision requirements for high-speed industrial manufacturing.

8.1 Summary of Contributions

- ****Tiling Inference Strategy:**** Developed a robust method for handling high-resolution industrial imagery without quality loss during resizing. - ****Optimization Pipeline:**** Demonstrated that an Optuna-driven, two-phase tuning process (Coarse-to-Fine) can significantly boost mAP performance on metallic surface datasets. - ****Validated Performance:**** Achieved a mAP@50 of 0.985 and a mAP@50-95 of 0.827, outperforming standard non-optimized baselines.

8.2 Directions for Future Research

1. **YOLOv11 Integration:** Future work should explore the recently released YOLOv11 and its enhanced transformer-based backbones to further increase sensitivity to low-contrast crazing.
2. **Semi-Supervised Learning:** Labeling industrial data is expensive. Implementing a semi-supervised or self-supervised framework (e.g., using masked autoencoders) could utilize the vast amounts of unlabelled data generated by production lines.
3. **Edge Optimization:** Quantizing the model to INT8 or FP16 for deployment on NVIDIA Jetson or similar low-power edge devices using TensorRT.

In conclusion, deep learning-based AOI systems are no longer a futuristic laboratory concept but a practical, ready-to-deploy solution for modern quality assurance.

Bibliography

- [1] Olatomiwa Badmos, Andreas Kopp, Timo Bernthaler, and Gerhard Schneider. Image-based defect detection in lithium-ion battery electrode using convolutional neural networks. *Journal of Intelligent Manufacturing*, 31(4):885–897, August 2019.
- [2] Paul Bergmann et al. Mvtec ad — a comprehensive real-world dataset for unsupervised anomaly detection. In *CVPR*, 2019.
- [3] Alexey Bochkovskiy et al. Yolov4: Optimal speed and accuracy of object detection. *arXiv preprint arXiv:2004.10934*, 2020.
- [4] A Ciampoli et al. Defect detection and characterization in rolled products. *Procedia Structural Integrity*, 24, 2019.
- [5] Tamás Czimmermann et al. Visual quality inspection and fine-grained defect detection: A survey. *IEEE Access*, 8, 2020.
- [6] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. In *NeurIPS*, 2021.
- [7] Alexey Dosovitskiy et al. An image is worth 16x16 words: Transformers for image recognition at scale. *ICLR*, 2020.
- [8] Stefan Elfwing et al. Sigmoid-weighted linear units for neural network function approximation in reinforcement learning. *Neural Networks*, 107, 2018.
- [9] Chengjian Feng, Yujie Zhong, Yu Gao, Matthew R. Scott, and Weilin Huang. Tood: Task-aligned one-stage object detection. August 2021.
- [10] Zheng Ge, Songtao Liu, Feng Wang, Zeming Li, and Jian Sun. Yolox: exceeding yolo series in 2021. 2021.

- [11] Zhora Gevorgyan. Siou loss: More powerful learning for bounding box regression. May 2022.
- [12] S. Ghorai et al. Automatic defect detection on hot-rolled flat steel products. *IEEE Transactions on Instrumentation and Measurement*, 62, 2013.
- [13] Ian Goodfellow et al. *Deep Learning*. MIT press, 2016.
- [14] Ian Goodfellow et al. Generative adversarial nets. In *NeurIPS*, 2023.
- [15] Kai Han, Yunhe Wang, Qi Tian, Jianyuan Guo, Chunjing Xu, and Chang Xu. Ghostnet: More features from cheap operations. November 2019.
- [16] Robert M Haralick. Statistical and structural approaches to texture. *Proceedings of the IEEE*, 67(5), 1979.
- [17] Robert M. Haralick, K. Shanmugam, and Its'Hak Dinstein. Textural features for image classification. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-3(6):610–621, November 1973.
- [18] Kaiming He et al. Deep residual learning for image recognition. In *CVPR*, 2016.
- [19] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(9):1904–1916, 2015.
- [20] Jonathan Ho et al. Riemannian denoising diffusion probabilistic models. In *NeurIPS*, 2025.
- [21] Qibin Hou et al. Coordinate attention for efficient mobile network design. In *CVPR*, 2021.
- [22] Glenn Jocher. Ultralytics yolov5. *GitHub repository*, 2020.
- [23] Glenn Jocher, Ayush Chaurasia, and Jing Qiu. Ultralytics yolov8, 2023.
- [24] Glenn Jocher, Ayush Chaurasia, and Jing Qiu. YOLO by Ultralytics. <https://github.com/ultralytics/ultralytics>, 2023. Accessed: 2023-01-19.
- [25] Glenn Jocher et al. Ultralytics yolo11. *GitHub repository*, 2024.

[https://github.com/ultralytics/ultralytics.](https://github.com/ultralytics/ultralytics)

- [26] Justin M Johnson and Taghi M Khoshgoftaar. Survey on deep learning with class imbalance. *Journal of Big Data*, 6, 2019.
- [27] Henning Kagermann, Wolfgang Wahlster, and Johannes Helbig. Recommendations for implementing the strategic initiative industrie 4.0. *Final report of the Industrie 4.0 Working Group*, 2013.
- [28] A. Kumar and G.K.H. Pang. Defect detection in textured materials using gabor filters. *IEEE Transactions on Industry Applications*, 38(2), 2002.
- [29] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521, 2015.
- [30] J. Lee et al. Hybrid-dc: A hybrid framework using resnet-50 and vision transformer for steel surface defect classification in the rolling process. *Electronics*, 13, 2024.
- [31] Chuyi Li, Lulu Li, Hongliang Jiang, Kaiheng Weng, Yifei Geng, Liang Li, Zaidan Ke, Qingyuan Li, Meng Cheng, Weiqiang Nie, Yiduo Li, Bo Zhang, Yufei Liang, Linyuan Zhou, Xiaoming Xu, Xiangxiang Chu, Xiaoming Wei, and Xiaolin Wei. Yolov6: A single-stage object detection framework for industrial applications, 2022.
- [32] Kunchang Li et al. Uniformer: Unifying convolution and self-attention for visual recognition. In *ICLR*, 2023.
- [33] Xiang Li, Wenhui Wang, Lijun Wu, Shuo Chen, Xiaolin Hu, Jun Li, Jinhui Tang, and Jian Yang. Generalized focal loss: Learning qualified and distributed bounding boxes for dense object detection, 2020.
- [34] Tsung-Yi Lin et al. Feature pyramid networks for object detection. In *CVPR*, 2017.
- [35] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. Microsoft coco: Common objects in context. May 2014.
- [36] Shu Liu et al. Path aggregation network for instance segmentation. In *CVPR*, 2018.
- [37] Yanqing Liu, Jianyang Gu, Kai Wang, Zheng Zhu, Wei Jiang, and Yang You. DREAM: Efficient dataset distillation by representative matching. In *Proceedings*

of the IEEE/CVF International Conference on Computer Vision (ICCV), pages 17314–17324, 2023.

- [38] Z. Liu et al. A survey of vision transformers in industrial inspection. *IEEE Transactions on Industrial Informatics*, 2023.
- [39] Ze Liu et al. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, 2021.
- [40] Quande Luo et al. Automated vision-based surface defect detection: A review. *Journal of Intelligent Manufacturing*, 31, 2020.
- [41] X. Lv et al. Deep learning for surface defect detection on industrial products. *IEEE Transactions on Industrial Informatics*, 2020.
- [42] E.N. Malamas et al. A survey on industrial vision systems, applications and tools. *Image and Vision Computing*, 21(2), 2003.
- [43] Daryl Powell et al. Zero defect manufacturing: A self-optimising perspective. *Computers in Industry*, 136, 2022.
- [44] F. Psarommatis et al. Zero-defect manufacturing: Current state and future trends. *International Journal of Production Research*, 2022.
- [45] Foivos Psarommatis et al. Zero defect manufacturing: state-of-the-art review, shortcomings and future directions in research. *International Journal of Production Research*, 58(1), 2019.
- [46] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: unified, real-time object detection, 2016.
- [47] Joseph Redmon and Ali Farhadi. Yolo9000: Better, faster, stronger, 2017.
- [48] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018.
- [49] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(6), 2017.

- [50] Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. *ICML*, 2015.
- [51] Karsten Roth et al. Towards total recall in industrial anomaly detection. In *CVPR*, 2022.
- [52] J.E. See. Visual inspection reliability for precision manufactured parts. *Human Factors*, 54(6), 2015.
- [53] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. September 2014.
- [54] Kechen Song and Yunhui Yan. A noise robust method based on completed local binary patterns for hot-rolled steel strip surface defects. *Applied Surface Science*, 285:858–864, 2013.
- [55] Petru Soviany and Radu Tudor Ionescu. Curriculum learning for two-stage object detection. *arXiv preprint*, 2018.
- [56] Xian Tao et al. Automatic metallic surface defect detection and recognition with convolutional neural networks. *Applied Sciences*, 8, 2018.
- [57] Zanjia Tong, Yuhang Chen, Zewei Xu, and Rong Yu. Wise-iou: Bounding box regression loss with dynamic focusing mechanism, 2023.
- [58] D.M. Tsai and C.Y. Hsieh. Automated surface inspection for statistical textures. *Image and Vision Computing*, 18, 2003.
- [59] Mihran Tuceryan and Anil K Jain. Texture analysis. *Handbook of pattern recognition and computer vision*, 1993.
- [60] Ashish Vaswani et al. Attention is all you need. In *NeurIPS*, 2025.
- [61] Ao Wang, Hui Chen, Lihao Liu, Kai Chen, Zijia Lin, Jungong Han, and Guiguang Ding. Yolov10: Real-time end-to-end object detection, 2024.
- [62] Chien-Yao Wang, Alexey Bochkovskiy, and Hong-Yuan Mark Liao. Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. July 2022.

- [63] Chien-Yao Wang, Hong-Yuan Mark Liao, I-Hau Yeh, Yueh-Hua Wu, Ping-Yang Chen, and Jun-Wei Hsieh. CspNet: A new backbone that can enhance learning capability of cnn. November 2019.
- [64] Jiaqi Wang, Kai Chen, Rui Xu, Ziwei Liu, Chen Change Loy, and Dahua Lin. Carafe: Content-aware reassembly of features. May 2019.
- [65] Sanghyun Woo, Jongchan Park, Joon-Young Lee, and In So Kweon. Cbam: Convolutional block attention module, 2018.
- [66] Haiping Wu et al. Cvt: Introducing convolutions to vision transformers. In *ICCV*, 2021.
- [67] L. Xie et al. A review of recent advances in surface defect detection using deep learning techniques. *Electronic Imaging*, 2020.
- [68] X. Xie. A review of classification algorithms for surface defect detection. *World Congress on Intelligent Control and Automation*, 2008.
- [69] Lingxiao Yang et al. Simam: A simple, parameter-free attention module for convolutional neural networks. In *ICML*, 2021.
- [70] Jiawei Yu et al. Fastflow: Unsupervised anomaly detection and localization via 2d normalizing flows. *arXiv preprint arXiv:2111.07677*, 2021.
- [71] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. *ECCV*, 2014.
- [72] Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization, 2017.
- [73] J. Zhang et al. Generative ai for industrial anomaly detection. *arXiv preprint*, 2023.
- [74] Y. Zhao et al. Few-shot defect detection: A survey. *Pattern Recognition*, 2024.
- [75] Zhaohui Zheng, Ping Wang, Wei Liu, Jinze Li, Rongguang Ye, and Dongwei Ren. Distance-iou loss: Faster and better learning for bounding box regression. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(07):12993–13000, April 2020.