COCO-Bridge: Common Objects in Context Dataset and Benchmark for Structural Detail Detection of Bridges

ERIC BIANCHI

Thesis submitted to the faculty of the Virginia Polytechnic Institute and State

University in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

In

CIVIL ENGINEERING

December 7, 2018

Blacksburg, Virginia

Keywords: Convolutional Neural Networks, Bridge Inspections, Drones, UAS, Deep Learning, A.I, Civil Engineering

COCO-Bridge: Common Objects in Context Dataset and Benchmark for Structural Detail Detection of Bridges

Eric Bianchi

ABSTRACT

Common Objects in Context for bridge inspection (COCO-Bridge) was introduced for use by unmanned aircraft systems (UAS) to assist in GPS denied environments, flight-planning, and detail identification and contextualization, but has far-reaching applications such as augmented reality (AR) and other artificial intelligence (AI) platforms. COCO-Bridge is an annotated dataset which can be trained using a convolutional neural network (CNN) to identify specific structural details. Many annotated datasets have been developed to detect regions of interest in images for a wide variety of applications and industries. While some annotated datasets of structural defects (primarily cracks) have been developed, most efforts are individualized and focus on a small niche of the industry. This effort initiated a benchmark dataset with a focus on structural details. This research investigated the required parameters for detail identification and evaluated performance enhancements on the annotation process. The image dataset consisted of four structural details which are commonly reviewed and rated during bridge inspections: *bearings*, *cover plate terminations*, *gusset plate connections*, *and out of plane stiffeners*. This initial version of COCO-Bridge includes a total of 774 images; 10% for evaluation and 90% for training. Several models were used with the dataset to evaluate model overfitting and performance enhancements from augmentation and number of iteration steps. Methods to economize the predictive capabilities of the model without the addition of unique data were investigated to reduce the required number of training images. Results from model tests indicated the following: additional images, mirrored along the vertical-axis, provided precision and accuracy enhancements; increasing computational step iterations improved predictive precision and accuracy, and the optimal confidence threshold for operation was 25%. Annotation recommendations and improvements were also discovered and documented as a result of the research.

COCO-Bridge: Dataset and Benchmark for Structural Detail Detection

Eric Bianchi

GENERAL AUDIENCE ABSTRACT

Common Objects in Context for bridge inspection (COCO-Bridge) was introduced to improve a drone-conducted bridge inspection process. Drones are a great tool for bridge inspectors because they bring flexibility and access to the inspection. However, drones have a notoriously difficult time operating near bridges, because the signal can be lost between the operator and the drone. COCO-Bridge is an image-based dataset that uses Artificial Intelligence (AI) as a solution to this particular problem, but has applications in other facets of the inspection as well. This effort initiated a dataset with a focus on identifying specific parts of a bridge or structural bridge elements. This would allow a drone to fly without explicit direction if the signal was lost, and also has the potential to extend its flight time. Extending flight time and operating autonomously are great advantages for drone operators and bridge inspectors. The output from COCO-Bridge would also help the inspectors identify areas that are prone to defects by highlighting regions that require inspection.

The image dataset consisted of 774 images to detect four structural bridge elements which are commonly reviewed and rated during bridge inspections. The goal is to continue to increase the number of images and encompass more structural bridge elements in the dataset so that it may be used for all types of bridges. Methods to reduce the required number of images were investigated, because gathering images of structural bridge elements is challenging,. The results from model tests helped build a roadmap for the expansion and best-practices for developing a dataset of this type.

ACKNOWLEDGEMENTS

A special thanks to the following sponsor

The National Science Foundation Center for Unmanned Aircraft Systems

A special thanks to the following companies for their donation of time and data:

Norfolk Southern, Roanoke

Clark Nexsen, Roanoke

AECOM, Roanoke

Michael Baker International

A special thanks to the following individuals and communities for their time, help, and expertise during the research project:

Mai Dahshan
Advanced Research Computing, Virginia Tech

Matthew Jezierski,
I.T. for the Department of Agriculture, Virginia Tech

Haseeb Chaudhry
PhD. candidate in Mechanical Engineering at Virginia Tech

The Virginia Tech Vision and Learning Reading Group

Alexa Nash

TABLE OF CONTENTS

CHAPTER 1: INTRODUCTION

1.1 Background

The American Society of Civil Engineers (ASCE) has stated, in their 2017 Report Card, that 40% of bridges are 50 years or older and 9.1% of all 614,387 bridges are structurally deficient (ASCE, 2017). It is estimated that the nation's backlog of bridge rehabilitation would cost approximately $123 billion. Engineers have been long practicing the art of designing bridges, however current needs require a substantial additional effort towards prolonging and protecting the safety of in-service bridges. Monitoring aging infrastructure continues to grow in importance as each year passes, since preventative maintenance action saves money and resources when proactively implemented (Otero, 2013).

As technology has advanced, there have been developments in the way bridge inspections are conducted in an effort to make them less subjective and obstructive, more detailed, and safer. One method of improvement is using drones to assist in bridge inspection. Drones are a way to avoid traffic obstructions, and navigate hard-to-reach areas, which may otherwise be difficult or require machinery for an inspector, However, flying unmanned aircraft systems (UAS) around bridges can be challenging because of the variable wind-speeds surrounding bridges, battery life of UAS, and loss of GPS from the amount of steel and obstructive size of bridges. Therefore, there has been a push toward the automation and semi-automation of future drone-inspection processes. Having autonomous on-board sensors and programming enables the drone to make decisions without the aid of a controller because of the GPS denied environment or variable wind conditions. UAS, like autonomous road vehicles, need to understand their environment, so they can make educated decisions when flying. Self-driving cars use a host of sensors including lidar and deep learning artificial intelligence (A.I) algorithms to understand their surroundings. Image processing techniques and A.I techniques such as using Convolutional Neural Networks (CNNs) help the car detect objects in its surroundings.

Modeling autonomous UAS by examining autonomous vehicles, means that the UAS will have on-board A.I. object recognition to assist in its flight. The capability to identify structural bridge details allows the drone to make decisions on where to fly next, or help in the localization of it position on the bridge. Flight planning will help with battery life optimization and autonomy during GPS denied conditions. Identifying structural details will also assist the inspector in focusing their attention on specific structural details and provides context to any defects identified. Ideally, the drone would be able to identify all the structural details and defects that an inspector would need to review, reducing the subjectivity of inspections by identifying consistently accurate and quantifiable metrics. The possibility of an autonomous bridge inspection, or A.I.-assisted inspection all begin with developing a structural detail dataset for bridges.

1.2 List of Contributions

This research has investigated the implementation of A.I. for UAS implementation through the following:

Development an annotated dataset of bridge structural details

The training Convolutional Neural Network (CNN) on the annotated bridge structural detail dataset

Comparison and evaluation of model performance of augmentation and overfitting risk

Built a proof-of-concept version of the bridge detection model for a drone

CHAPTER 2: BACKGROUND AND LITERATURE REVIEW

2.1 Standard Bridge Inspection Process Overview

Inspection Process

The most common form of bridge inspection is the visual inspection performed by trained inspectors. State DOT agencies base repair/rehabilitation decisions upon data obtained from inspections. (Otero, 2013). Regular inspections must occur once every two years, or on an as needed basis. There are different overall bridge inspection assessments, as well as inspections to specific condition states of individual structural members. The lowest tier of inspection is the security inspection. It checks for obvious defects from vehicular damage or vandalism. The next level is general inspection which is a physical condition assessment. This form of inspection is done every two years and does not impair traffic much. The most extensive inspection, the principal inspection, involves close examination and hands-on instruments. There may also be special inspections which involve a particular element or area, such as a fractural critical element (Hüthwohl, Lu, & Brilakis, 2016). A fracture critical bridges may have one or more fracture critical elements. These elements are certain details specific to the bridge or design that if fractured, run the risk of the bridge failure or catastrophic failure.

Once a defect has been found, the inspector must determine the cause. A book of defect images can be used to assist an inspector to gage the extent or rating of the damage. The physical properties are recorded so that the defects may be monitored for worsening conditions. The types of defects found in bridges are outlined in the next section. Inspectors must find any new defects as well as monitor the existing defects found previously. Most inspections are conducted visually, but in some cases require hands-on tools to gage the condition (Hüthwohl et al., 2016).

Methods of Inspection: Details and Defects

Inspectors must consider the entire structure, but also must pay attention to specific connection details in a bridge to ensure that these locations are structurally sound. An inspector may find a defect in the detail, and it will be their job to document and determine its significance. Table (2.1.1) includes typical defects which are considered when inspecting bridges. Table (2.1.2) includes information on typical details in bridges which come with structural contingencies which must be considered. In the Bridge Inspector's Reference Manual (BIRM), each type of inspection procedure (Box beam, T-beam, etc.) that "the inspection for cracks, spalls, and other defects in concrete bridges is primarily a visual activity (NHI, 2002). However, hammers and chain drags can be used to detect areas of delamination." Cracks are particularly difficult to quantify in inspections since the measurements can be subjective (Hüthwohl et al., 2016). It also notes advanced techniques for inspection such as ground-penetrating radar and ultrasonic testing (NHI, 2002). Other advances in concrete bridges such as using thermal imagery to show cracks and delamination have been proven to be successful in bridge inspections (Mohan & Poobal, 2017). In steel bridges, like concrete bridges, it is mostly a visual inspection to identify defects, until a tool or instrument is used to further identify the defect.

Table 2.1.1 - Typical Defects Found in Bridges (NHI, 2002)

Timber	Concrete	Steel
Fungus Decay	Cracking	Bent, damaged, missing members
Damage by parasites	Scaling	Corrosion
Deflection	Delamination	Fatigue Cracks
Checks	Spalling	Other stress-related cracks
Splits	Efflorescence	
Shakes	Honeycombs	
Loose connections	Pop-outs	
Surface Depressions	Wear	
Chemical Attack	Collision damage	
	Abrasion	
	Overload damage	
	Reinforcing steel corrosion	
	Prestressed concrete deterioration	

Fatigue in steel is a major problem which limits the load-carrying capacity and life of an existing steel structure (Haghani, Al-Emrani, & Heshmati, 2012). Having well-planned inspections which identify these fatigue-prone structural details, along with successful repair schedules helps ensure structural integrity through a bridges service life. Because this paper is focusing on examining steel bridges, only a reference table (Table 2.1.2) of the types of details which are prone to fatigue cracking in steel bridges was created. This reference table was created from structural details found in the BIRM manual. Figure (2.1.1) was a chart on the frequency of certain fatigue-prone details collected from a research article (Haghani et al., 2012).

Table 2.1.2 - Fatigue Prone Details in Steel Bridges (NHI, 2002)

Fatigue-Prone Details	Tri-axial constraints
	Intersecting Welds
	Cover-Plates Terminations
	Beam Seats
	Field Welds
	Intermittent Welds
	Transverse Stiffener (beam web)
	Web Gap at End of Transverse Connection Plate (8.1.40)[1]
	Web Gap at Floor beam Connections to Main Girders
	Flange Termination

[1] Out of Plane Bending occurs at two common locations: Web gaps at end of transverse connection plates and web gaps at floor beam connections to main girders. This kind of problem could be present in truss bridges, suspension, two-girder, multi-beam/girder, tied arch, and box girder bridges.

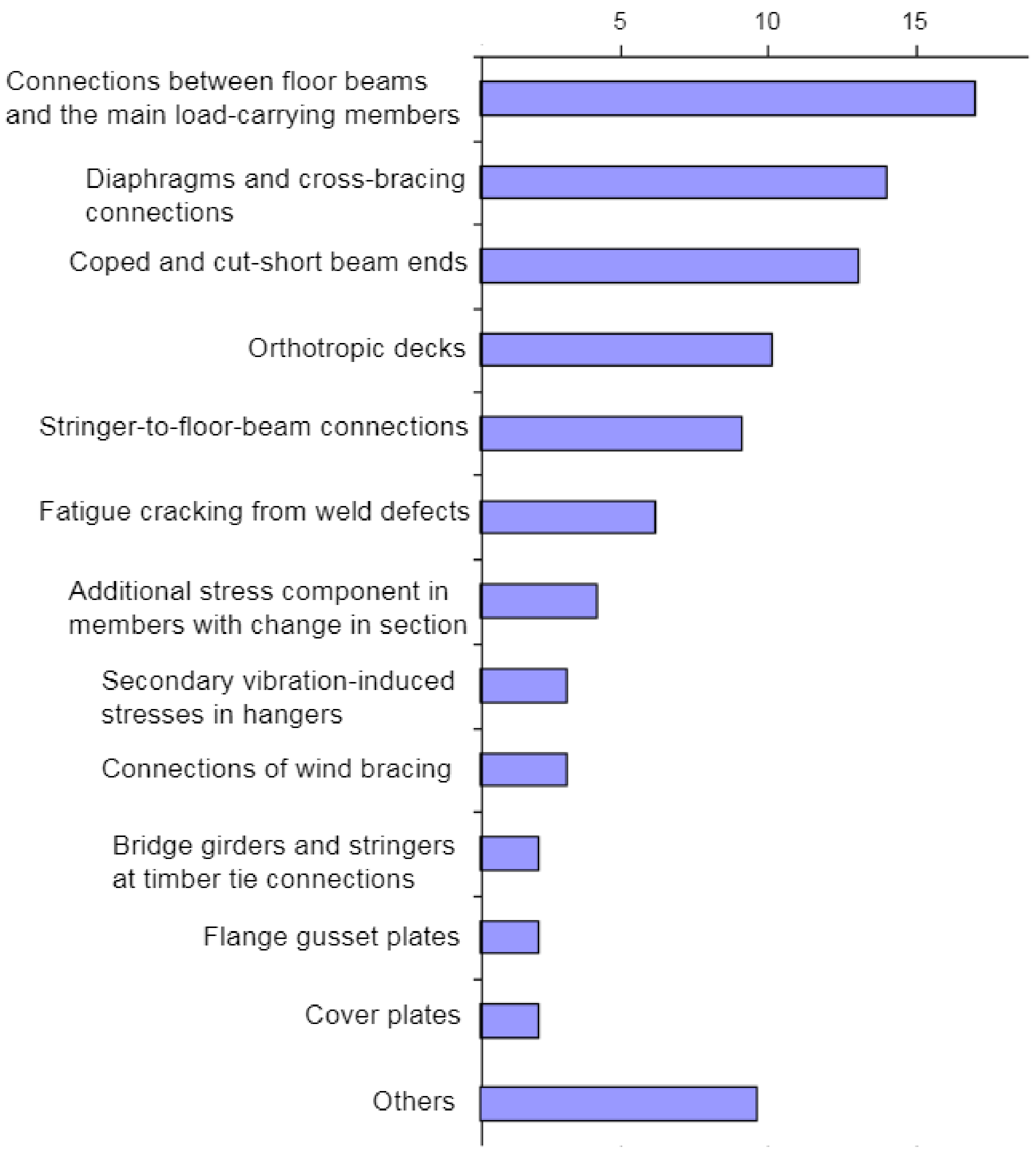


Figure 2.1.1 - Frequency of Fatigue Damage Cases

Challenges and Limitations

Challenges and limitations in bridge inspections derive from safety, traffic obstruction, the human element of subjectivity, time, and bridge's hard-to reach areas. Bridge inspections can be hazardous for the people involved, requiring inspectors to do their job in unfavorable conditions. There are two common types of bridge inspection access methods, Aerial Work Platforms (AWPs) and Rope Access. AWPs are the most common method for accessing difficult to reach areas of a bridge (Lovelace, 2015). They use vehicles such as snoopers, lifts, or bucket-trucks. This method is common, reliable, and the inspector is an arm’s length away from the bridge. However, it is costly, impairs normal traffic, and is an added safety hazard to the inspectors. Rope Access requires certified rope professionals. This method is used when there are portions of the bridge not accessible from the bridge deck or the ground. It is low cost, does not impair traffic, and the inspector is on-site. However, safety hazards, availability, and training are required to conduct this method of inspection. Humans are subjective can be inconsistent. Inconsistencies from one inspector to another inspector is something that the industry tries to protect against. The check in 2008 for the United States was for three bridge inspectors to examine an identical bridge independently. Their results are compared and if they vary significantly, they may lose their certification (Hüthwohl et al., 2016).

2.2 UAS Bridge Inspection Overview

UAS Inspection Process Research

UAS are an emerging approach to bridge inspection. UAS assisted real-time inspection may allow the inspector to view the bridge from a screen. The inspector would avoid walking around, doing roping, or the use of any kinds of rigging to view the bridge (Hüthwohl et al., 2016). LSER Labs tested UAS assisted inspections on various bridges, wood, concrete and steel. On the steel bridge, their inspection examined gusset plates and found a common defect of missing nuts and bolts. There were also substantial rust and corrosion and certain areas that were identified. The drones were able to stream video with a level of clarity to 0.16cm. Although AASHTO guidelines states that the minimum crack size that inspectors must include in reports are cracks with a gage length of 0.01cm, the researchers were able to record image data which met the ASHTO level of accuracy (Otero, 2013). Another use of UAS for bridge inspection is through Building Information Modeling (BIM). This method is where the drone collects bridge data through means of 3D point clouds and/or images. A report by researchers at the Oregon University goes into extensive detail of using drones for bridge inspection (Gillins, D; Parrish, C; Gillins, M; Simpson, 2018). Multiple tests were conducted using drones for in-flight 3D modeling of bridges and steel signal towers. Their process involved manual and pre-determined flight patterns for the structures observed. Inspectors can use these 3D models to reference the state of a bridge at a later point in time. Figure (2.2.1) is an example 3D point cloud bridge model made in the Oregon report by overlapping 2D imagery (Gillins, D; Parrish, C; Gillins, M; Simpson, 2018). The report also examined the potential for cost savings by implementing these techniques. Another paper examined how geo-referenced, high-dense point clouds, captured by drones using the software CloudCompare could be used for change detection. The implementation was used to monitor a retaining wall (Figure 2.2.2) (Hallermann, Morgenthal, & Rodehorst, 2015). These models could be potentially used to re-create the bridge in a 3D space to record and detect any defects which may be present (Hallermann et al., 2015).

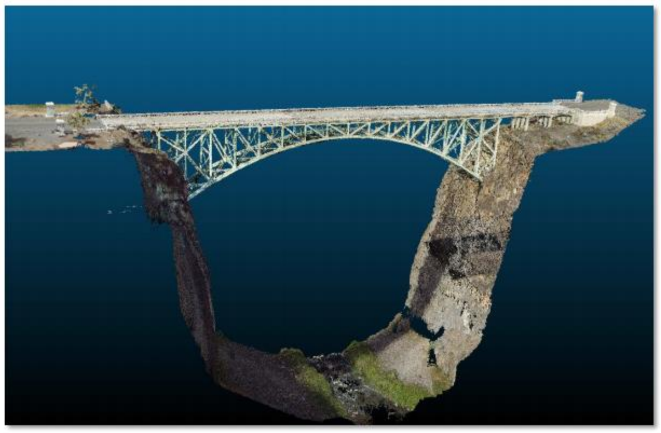


Figure 2.2.1 - 3D Point Cloud of Crooked River Bridge (Gillins et al., 2018)

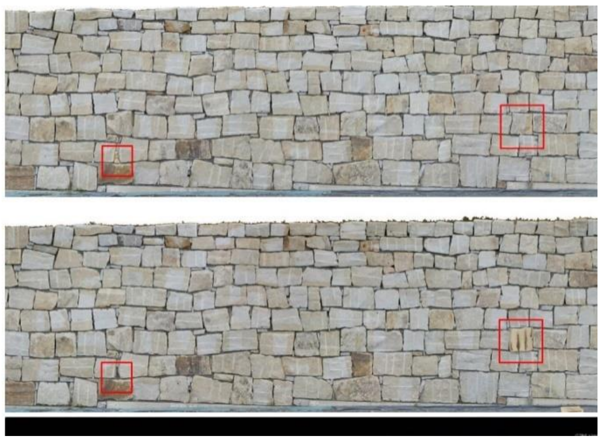


Figure 2.2.2 - Georeferenced Point Cloud of Retaining Wall to Detect Change (et al., Hallermann , 2015)

Feasibility

UAS inspection has shown cost-benefit potential but is limited by several factors: the bridges it effectively can be used on; GPS signal loss; and regulations. Preliminary costs savings were estimated by the TSER Labs study. Their cost parameters included an operator, equipment, maintenance, repair, and video editing costs. The expected cost savings would come from the reduced on-site staff, and initial equipment costs would be covered from just 1-2 inspections. The work focused on the actual flight of the drones and feasibility of the UAS for bridge inspections. One of the recommendations for the future is to better-understand the over-all mission dynamics when using drones to collect images/video data of an entire bridge (Otero, 2013). The report from Oregon University analyzed potential cost savings over $10,000 where it was a UAS suitable project (Gillins, D; Parrish, C; Gillins, M; Simpson, 2018).

Minnesota DOT did a case study in 2015 on the feasibility of bridge inspections using Drones. They tested a drone's effectiveness and feasibility on several bridges of varying length and composition to get a high-level overview of its performance. The conclusions from these tests were that bridges which were hard to access were great candidates for drone supplementation. However, the limiting factor was that there were losses of signal and GPS which made flying underneath bridges difficult (Lovelace, 2015). Another conclusion was that for short-span small-height bridges, drones were found to not be as ideal (Lovelace, 2015). Among some other reasons, short-span small-height bridges as not an ideal candidate for drones since these inspections can be easily conducted on-foot.

Regulations and UAS inherent limitations have curbed the commercial use of drones in bridge inspections. Current laws do not allow for UAS to fly over traffic stating: "You cannot fly a small UAS over anyone who is not directly participating in the operation" (FAA, 2018). Regulations like these have inhibited the use of drones in bridge inspections. However, one area of acceptable UAS use are traffic devices, signs, and high-mast lighting. If these structures are not over traffic, these become ideal candidates for drone inspection, according to Staci Dugan, a senior inspector from Clark Nexsen. Although drones have the potential to reach areas which would be difficult for humans to reach, there are also areas which are difficult for drones to reach. There are some things which drones have historically had trouble doing, like inspecting the underside of a bridge (Lovelace, 2015).

Before going to a site, it seems that there will need to be some criteria made on whether drone-flight is appropriate. MDOT conclude that drones provide important pre-planning for large-scale inspections. As quoted from Staci Dugan, "The most difficult part of a bridge inspection is determining what tools to use". Drones, at the moment, are not a generic tool for bridge inspections, rather they are a tool which may serve specific purposes (Gillins, D; Parrish, C; Gillins, M; Simpson, 2018) (Lovelace, 2015). The Minnesota

Bridge Inspection paper explains that drone use would be best for bridges which do not require hands-on inspection. MDOT concludes that UAS are more suitable as a tool to assist, but they themselves cannot perform inspection independently. The example the document uses where drones would be useful would be to observe from some distance much greater than arm's length on bearings or connections. Drones are something which may aid to the human inspection process. An agreeable position from Staci Dugan was that drones may supplement and enhance a bridge inspector's experience, but not replace it.

2.3 Artificial Intelligence and Neural Networks

Artificial Intelligence

John McCarthy was one of the founders of artificial intelligence, and he defines artificial intelligence as “the science and engineering of making intelligent machines” (Skymind, 2017). Another definition is: a computer system able to perform tasks that normally require human intelligence, such as visual perception, speech recognition, decision making, and translation between languages. *Machine learning* is a subset of artificial intelligence (Skymind, 2017). Machine learning was defined by Arthur Samuel, one of the forerunners of machine learning in 1959 as “a field of study that gives computers the ability to learn without being explicitly programmed”. Machine learning programs become *smarter* by optimizing a certain dimension, called an *objective function*, to minimize error. Deep learning is a subset of machine learning. Figure (2.3.1) describes the relationship between artificial intelligence, machine learning, and deep learning.

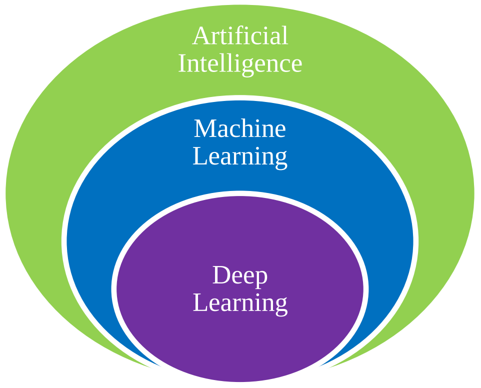


Figure 2.3.1 - Hierarchy of Artificial Intelligence

The term *deep* refers to the number of hidden *layers* in a *neural network*. Neural networks cane be defined as: a computer system modeled on the human brain and nervous system (University of Wisconsin, 2007). Layers can be thought of as a number of interconnected nodes which contain activation functions. The input layers communicate patterns to the hidden layers. Inside these hidden layers is where the processing is done through weighted connections (University of Wisconsin, 2007). The output layer consists of nodes which represent the number of outcomes defined. For example, the network could output a true or false statement, in which there might be two nodes. The following image is a visual representation of a neural network.

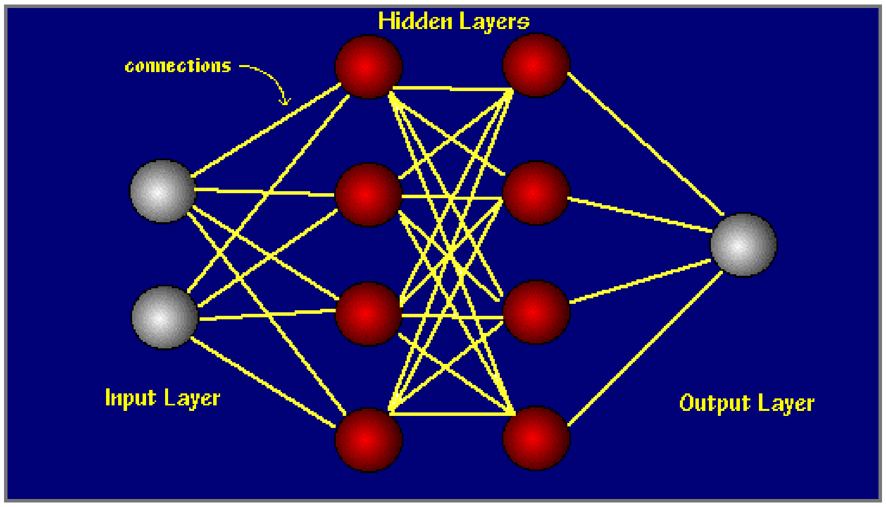


Figure 2.3.2 - Neural Network Visualization (University of Wisconsin, 2007)

Convolutional Neural Networks

Convolutional neural networks (CNNs) were inspired by the biological visual cortex. When humans see something, layers of neurons detect lines, colors, and complex features and relations to perceive what is being seen. In comparison to feedforward neural networks, with similarly sized layers, CNNs have fewer connections and parameters making them easier to train while sacrificing minor performance losses. CNNs have been able to give record-breaking results (Krizhevsky, Sutskever, & Hinton, 2012) on image processing and other complex problems. Some popular neural networks are the Single Shot Detection (SSD) (Liu et al., 2016), You Only Look Once (YOLO) (Redmon, Divvala, Girshick, & Farhadi, 2015), and GoogLeNet (Szegedy et al., 2014). Convolutional neural networks are made up of layers, like its name suggests. The more layers a network has, the *deeper* the neural network. The three main types of layers are convolutional layers, max-pooling layers, and fully-connected layers. These layers may be repeated a number of times depending on the depth of the designed architecture. This report will call these three layers the CNN stack.

Convolutional Neural Networks Architecture

A CNN's architecture can be broken down into an input layer, CNN stacks, and an output layer. The input image layer is defined as the volume of the image. It is a volume because there is an image height, width, and depth (Red, Green, Blue or RGB) (Cha, Choi, & Büyüköztürk, 2017). For example, an image pixel size of 5 x 5 would actually be read as 5 x 5 x 3 since there is an RGB depth to the image. Then follows the CNN stacks.

The first layer would be a convolutional layer, which performs dot product multiplication between a *receptive field* and a subarray of an input or a filter. The filter is an array of number weights which convolves or slides across the image and extracts a single value through dot product multiplication for the given position (Figure 2.3.3). This results in a single integer value which comprises the output volume, convolved feature, or *feature map*. The action of the output *feature maps* is obtained by summing one or more convolutional responses that are passed through a pixel-wise *activation function* (Zhou, Chen, Zhang, Xie, & Zhou, 2016). The sub-array filter is then slid over to compute the next receptive field. This process is repeated until the entire image has been processed.

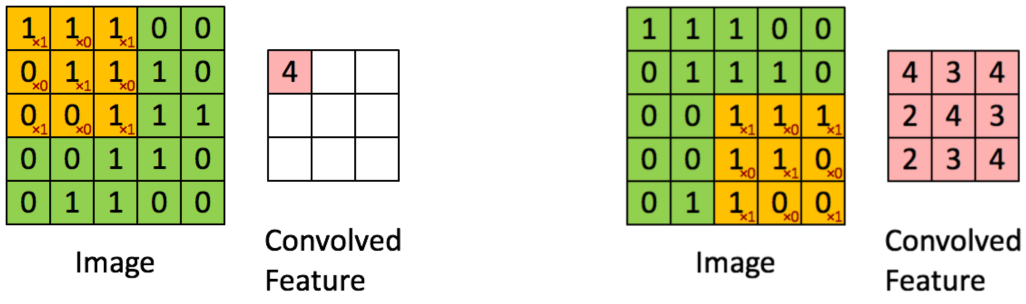


Figure 2.3.3 -Convolution Layer, Beginning on Left, Finished on Right (Stanford, 2013)

An *activation function* is used to determine the output of neural networks and maps the resulting values between 0 and 1. A popular, widely used activation function is the nonlinear Rectified Linear Units (ReLUs) function. This function improves the model's gradient descent, or how fast the function converges (Krizhevsky et al., 2012). This model trains several times faster than other nonlinear functions such as tanh or sigmoid (Cha et al., 2017).

The next layer in the stack is a pooling layer, which performs a function to reduce the computational complexity of a model by reducing its matrix dimensions. Essentially, it speeds up the computational training process at the expense of losing information. One common function is max-pooling, which takes the “most important” part of a certain part of the input volume. Pooling layer helps the model become invariant against rotations in the image. It is important to note that there can be several convolutional layers before a pooling layer. The max pooling process can be visualized through the following Figure (2.3.4).

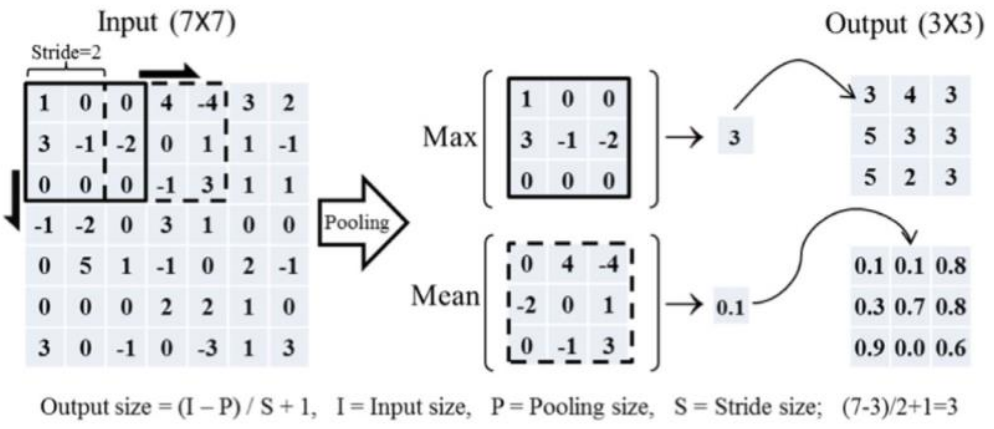

Input (7X7)
Stride=2
Pooling
Max
Mean
Output (3X3)
Output size = (I – P) / S + 1, I = Input size, P = Pooling size, S = Stride size; (7-3)/2+1=3

Figure 2.3.4 - Pooling Layer Representation (Cha et al., 2017)

The fully connected layer connects every neuron in one layer to every neuron in another layer (Krizhevsky et al., 2012), and the last fully connected layer may use softmax activation function instead of the ReLU. This softmax activation, in the last fully connected layer, predicts the type of *class* from the input data. A class can be thought of as an outcome option, like a prediction percentage that the image is of a dog, a cat, or a human. In terms of matrices, the fully connected layer has an input volume from the previous layer and outputs a vector size of ‘n’ number of classes. If the network were detecting dogs, cats and humans then the vector size would be 1 x 3. The vector would contain the probabilities for each these classes.

When a CNN is being trained, it is being fed image data with corresponding tags or labels to correspond to its classes. The model learns what features are most important to specific classes through backpropagation process by adjusting its filter weights. The backward pass refines these weights so that the error loss function decreases and updates the weights. Error loss may be affected by the pre-defined learning rate. The learning rate is the speed at which the model converges, often defined as an error loss parameter, and may be adjusted in the initialization of the model.

2.4 Levels of Automating Detection

Civil engineers have researched and put into practice methods of automating of defect detection. It was conceived that there were three major categories of detection. The first is identifying that there is an issue. The second is quantifying the extent of the defect. The third level assumes that there has been a baseline established and compares the current damage to that baseline. One other method of detection, which falls outside of the classical detection methods used by bridge inspectors, is predictive models of deterioration rates. Predictive models are the detection of defects before they occur. There have been many methods employed for detection, and this paper will explore several.

Classification and Object Detection

There are two main options of input data for structural defect detection that other authors' have developed in previous research. The first is using 2D images, and the other is through 3D point cloud models. For 2D images, there have been numerous papers which have referenced this input data (***list***). Some papers focused on edge detection, or color change. For example, one of the papers focused on the change in pixels of 2D images (Guldur & Hajjar, 2016) to detect cracks. The scripts were written in Matlab and utilize OpenCV edge detection algorithms as well as extensive procedures to isolate the crack detection. The proposed method struggled with water-marks, shadows, and cracks which were wide or light in color. Recently more papers have been using more modern image processing techniques involving machine learning. Convolutional neural networks have shown success in being able to detect cracks in an image (Cha et al., 2017). A paper which involved the detection of cracks in concrete used a method of image classification, where an image was taken from a camera and broken into much smaller images. These images were classified as either cracks or not cracks. If it was a crack, then a box was drawn around the target area. If it did not detect a crack, then no box was shown. This effectively highlighted the region where cracking was occurring with measured success. This training model only used 277 images, with 55 for validation. This form of detection can be seen in the following Figure (2.4.1).

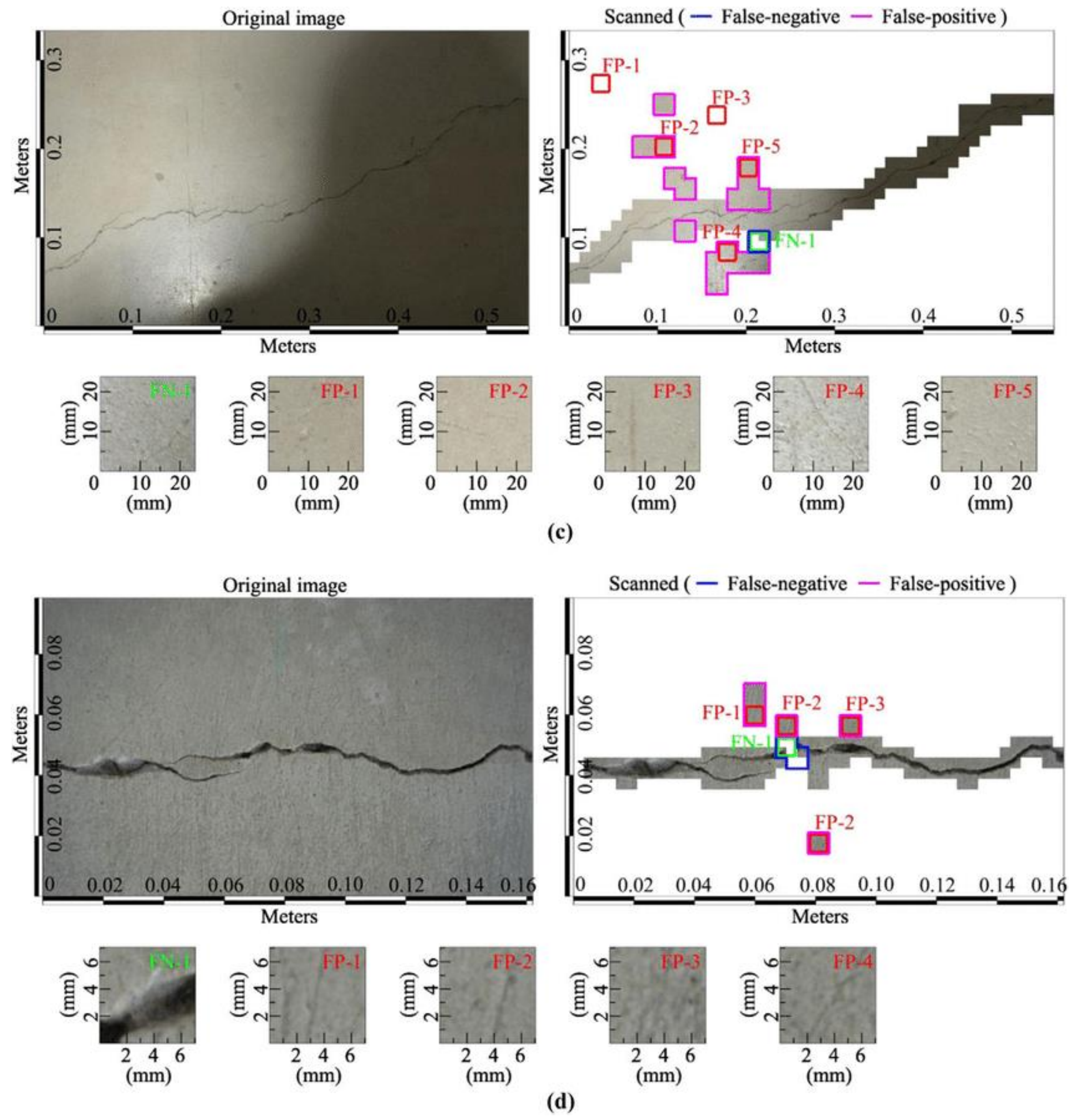
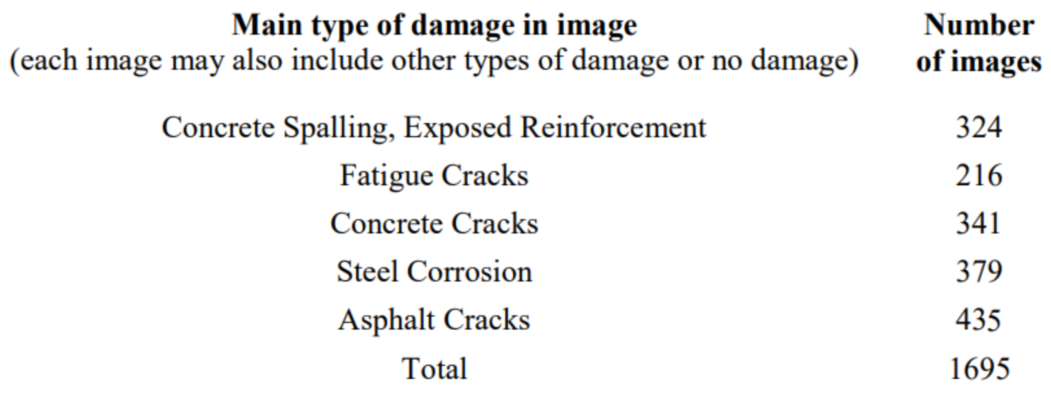


Figure 2.4.1 - Convolutional Neural Network Crack Detection (Cha et al., 2017)

A paper from the University of Illinois was on vision-based structural inspection using multi-scale convolutional neural networks (Hoskere, Narazaki, Hoang, & Spencer, 2017). They trained their model with approximately 1,600 images for 5 classes. The end result presented a colorful visual of the defects in each of the images in Figure (2.4.3). However, the low number of images used would limit the model's effectiveness in robust generalization across new environments.



Main type of damage in image (each image may also include other types of damage or no damage)	**Number of images**
Concrete Spalling, Exposed Reinforcement	324
Fatigue Cracks	216
Concrete Cracks	341
Steel Corrosion	379
Asphalt Cracks	435
Total	1695

Figure 2.4.2 - Trained Classes and Corresponding Number of Images Trained (Prasanna et al., 2016)

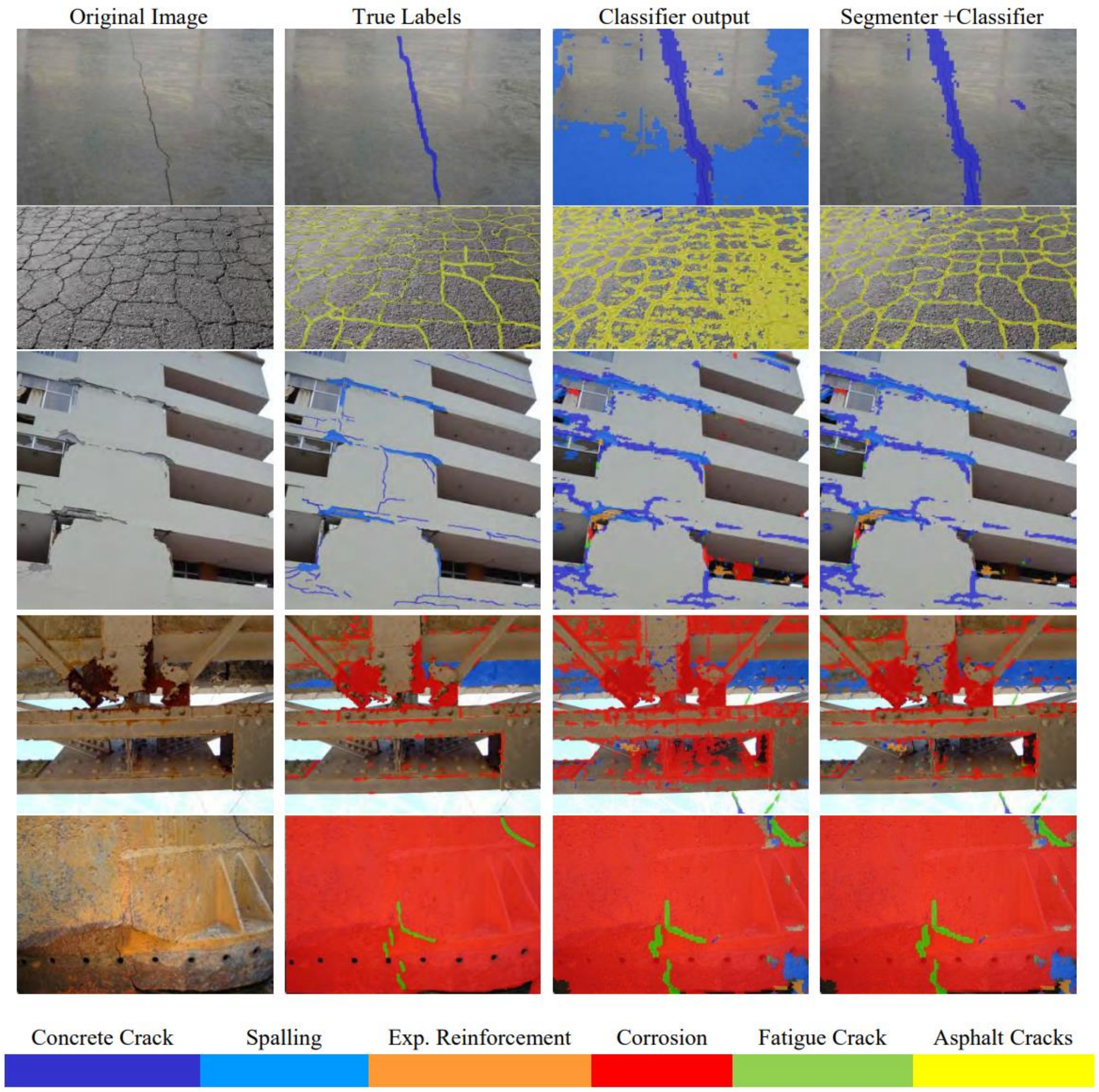


Figure 2.4.3 - Defect Detection with Pixel-level Colored Identifiers (Hoskere et al., 2016)

An on the ground robot was developed to detect cracks through the partially tuned robust multi-feature classifier (Prasanna et al., 2016). It recorded a peak performance of 95%, utilizing localized line segments, a spatially tuned multi-feature computation, and a machine learning image classifier to develop a local crack map.

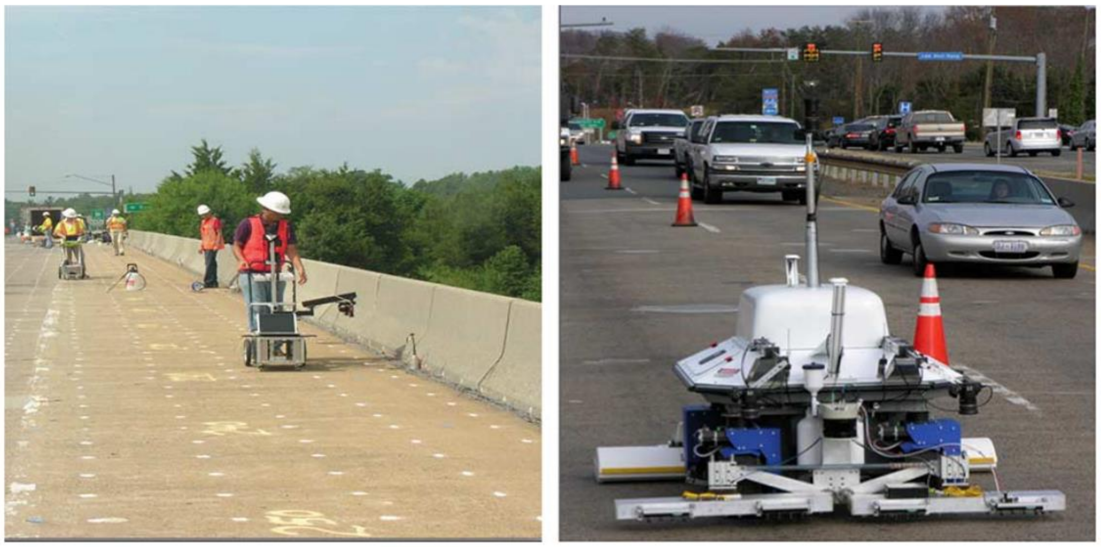


Figure 2.4.4 - Automatic Robotic Crack Detection (Prasanna et al., 2016)

A paper at Purdue presented a novel approach at defect detection in bridge inspection by identifying defect-susceptible areas on an image to narrow the scope which the algorithm searched (Yeum, 2016). The regions of interest (ROI) were extracted and vision-based visual techniques were used on the ROIs. Since looking over entire images sometimes creates "false-alarms", especially on steel structures due to its homogenous nature, this helps to narrow the scope which an algorithm would need to search for defects. If the detection was localized to susceptible areas, it would reduce that chance of false-readings (Yeum, 2016). Failure sensitive regions were detected on the test structure through image classification training. The detector boxed in ROIs which would be susceptible to defects Figure (2.4.4).

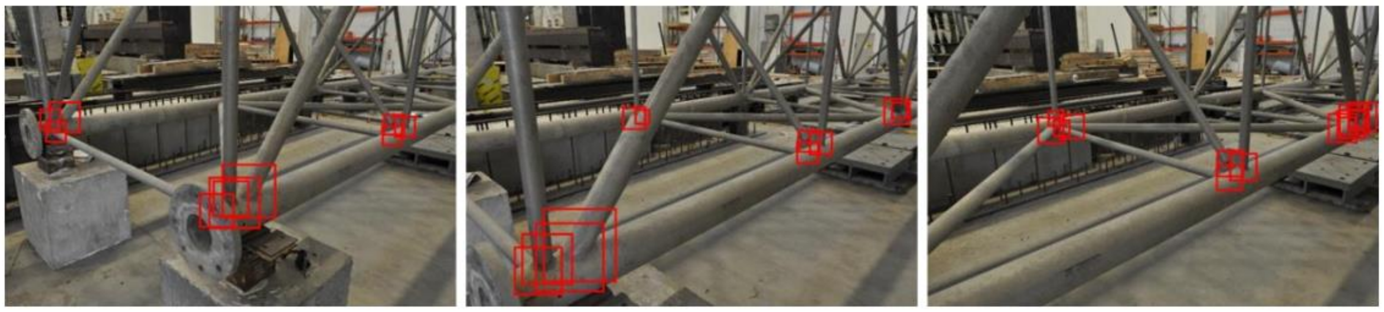


Figure 2.4.5 - Region of Interest Detection (Yeum, 2016)

For 3D images there are two methodologies for obtaining data. The first is through a surface-normal based damage detection and the second is a graph-based object detection method (Guldur & Hajjar, 2016). A report used neural networks to determine where there was significant damage on a laser-scanned structural frame (Guldur & Hajjar, 2016). This method proved to give accurate results for their test. In the conclusion, that authors acknowledge that surface normal-based damage detection from images for cracks and spalling can detect but not quantify damages. In this paper they only used 201 damage and non-damage vectors. It was suggested that further research should examine larger datasets with varying surface properties. Another report used CNNs to automate pavement crack detection with pixel-perfect accuracy (Wang, Allen, Yang, Cheng, & Gary, 2017). *Pixel perfect* accuracy is a muddled definition and can mean different things. The most common meaning of *pixel perfect* comes from web design where the pixels do not look blurry or contorted from scaling (Brett, 2016). In this definition, the author is referring to how the detected features are down to the pixel level, undistorted by the pooling layer. It was a novel approach without the use of the common pooling layer in the CNN architecture. The model was trained with 1,800 3D pavement images and showed successful results returning 90% precision on its 200 3D image training set.

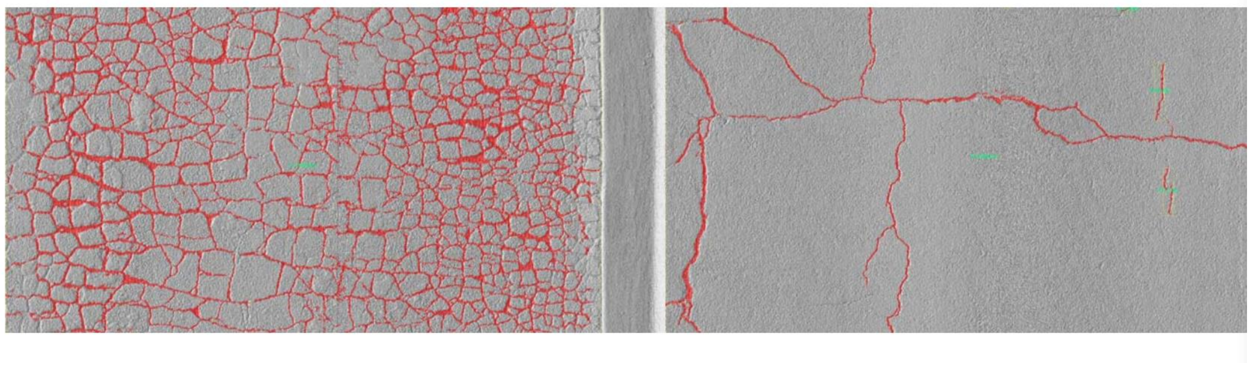


Figure 2.4.6 - Automatic Pixel-level Crack Detection (Wang, et al., 2017)

Quantification Detection

Common ways of quantifying the extent of a defect using an image is through the pixel area of the defect or pixel length of the defect. This is done by utilizing 3D point cloud image reconstruction techniques (Mohan & Poobal, 2017). Lasers give an accurate measurement of distance from the camera to the element. An example of using 3D point clouds was a report written by Zheng at Virginia Tech which explains how to go from *point cloud space* (pixel space) to *measurement space* (Zheng, Moen, Roberts-Wollmann, & Cousins, 2014). The percent error from the true crack widths were under 10%. The procedure given in the report was to first locate the crack, zoom in on the location, determine where on the structure the crack was located and measure the crack from color change to color change. The limitation was that it was found to be computationally expensive and the error was between 5-10%. One of the conclusions in the paper was that it would be beneficial to automatically detect cracks and have it automatically determine the width.

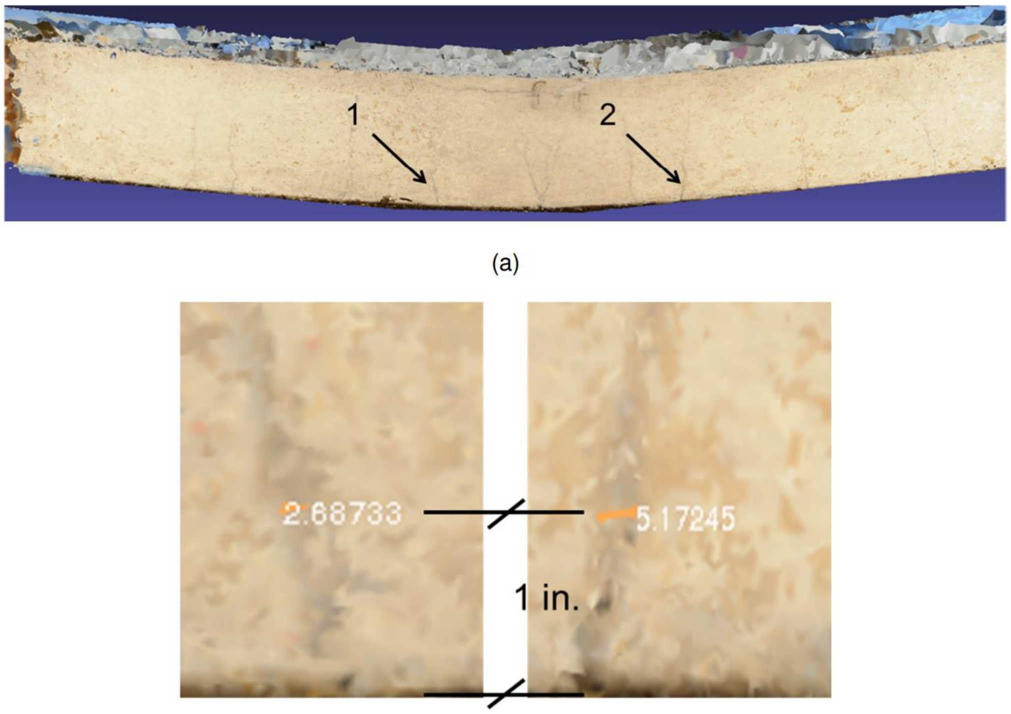


Figure 2.4.7 - Quantifying Crack Widths Using a 3D Point Cloud (Zheng et al., 2014)

A bridge crack detection and quantification method using UAS was developed out of Taiwan (Rau et al., 2016). The researchers used a UAS to create a 3D model from 2D images using Agisoft photoscan and a local coordinate system was established using ground control points. Defects such as cracks were automatically detected using an object-based image analysis technique to segment the image into objects. Then they applied semi-global-matching to obtain 3D crack information based on surface geometry to scale the image to calculate widths. They developed their own GUI interface for crack evaluation seen in Figure (2.4.8). In areas of spalling they were able to quantify the damage through volume calculations. They produced accuracies over 90% for their case studies, and were interested in change analysis for the future.

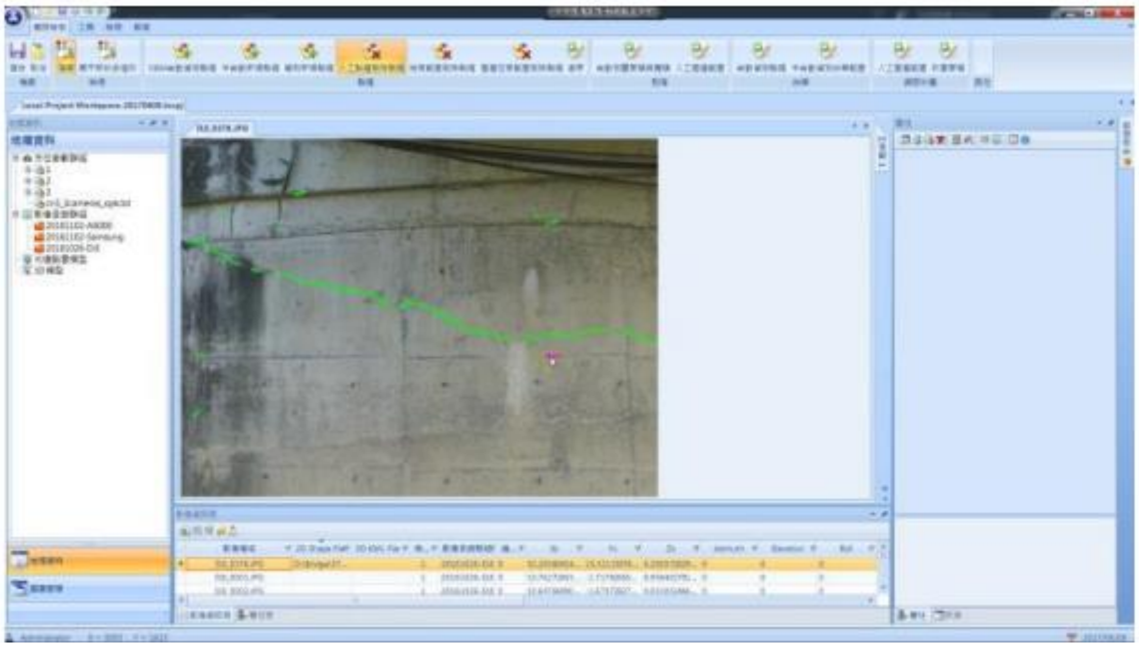


Figure 2.4.8 - Automatic Crack Detection Interface for Quantification of Cracks and Spalling (Rau et al., 2016)

Baseline Change Detection

It is beneficial to detect a defect in infrastructure, but it is superior to detect change in that defect over time. One paper explored the challenge of detecting defect change over time using a moving camera with 2D images (Stent, Gherardi, Stenger, & Cipolla, 2015). The inspection setting was a tunnel, and the research acknowledged the effect of change in lighting and registration error. The research goes onto discuss the use of a two-channel convolutional neural network to detect change. The system looks into “stage 4” (Figure 2.4.9) of the change detection process outlined in the research paper. The neural network is trained to learn what the same-image variation should look like, so that it may detect the abnormal changes.

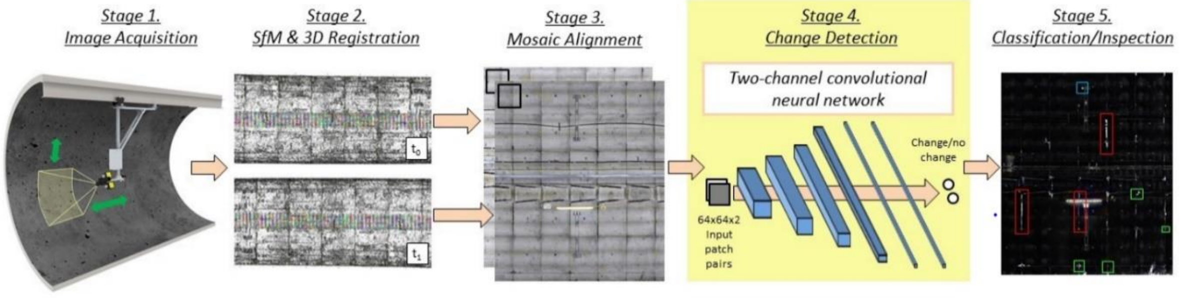


Figure 2.4.9 - Change Detection Over Time (Stent et al., n.d.)

Detecting change is difficult because of scale of the changes over time, nuisance factors, registration error. The scaling of cracks, for example, are typically small and hard to quantify. Nuisance factors such as change detected may be misinterpreted because of lighting, or the surface being wet. Finally, there is registration error which is achieving the pixel-accurate registration. The proposed system captured viewpoints from multiple angles and from different cameras from the same time (Stent et al., 2015).

Another proposed way to detect change is through a 3D point cloud using a baseline method (Shen, Lindenbergh, & Wang, 2017). However, detecting change in a 3D point cloud is challenging due to the errors from registration. One way to avoid these errors is to establish baselines as virtual points from a reconstructable feature in the scene used. In essence, the point cloud becomes a structural coordinate system which identifies relative motion.

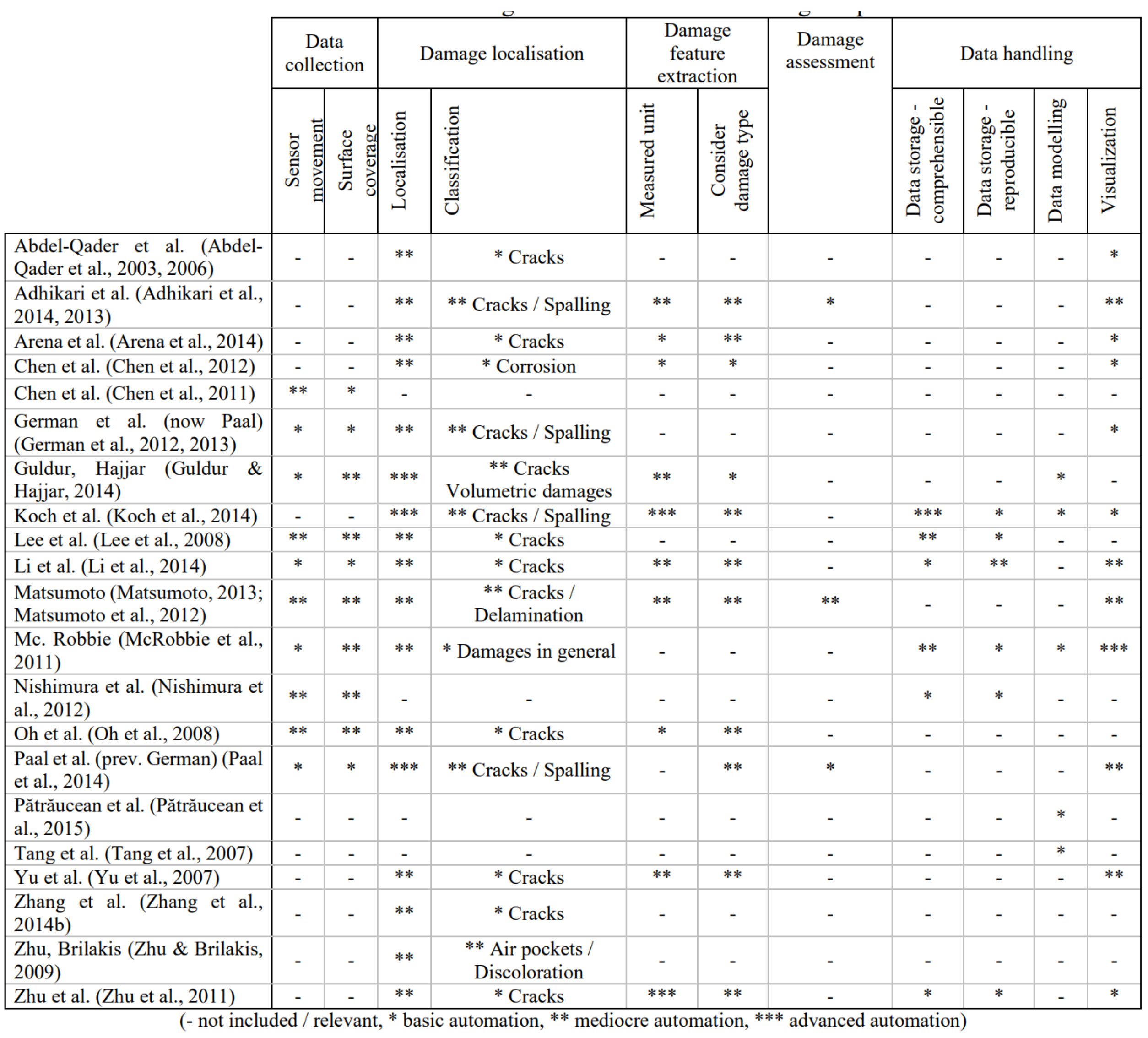
Predictive Deterioration

There has been some effort into predicting the extent of defects in bridges. Most models consider the design life of bridges rather than the deterioration of a single area of interest (Abu-Tair, Mcparland, Lyness, & Nadjai, 2002) (Pitt&Sherry, 2015). Predictive models consider many variables and time-history information. This information could be used in the form of big data to create predictive design life estimates.

General Summary of Findings

Whether the detection is classifying, quantification, or predictive and the method is through 3D point clouds or 2D images, it was found that most existing methods are addressing only one damage type, mostly cracks (Hüthwohl et al., 2016) (Table 2.4.2). Another issue is that the sensor data does not consider structural geometry. Thus, it can identify the defect, but not its implication, "the assessment of bridge elements based on extracted damage properties is largely unexplored" (Hüthwohl et al., 2016). The following is a table summarizing papers which investigate crack detection, compiled by (Hüthwohl et al., 2016). This table, while outdated, does still provide a good overview of the defect detection, and quantification research.

Table 2.4.1 - Overview of Existing Methods for Automated Bridge Inspection (Hüthwohl et al., 2016)



	Data collection		Damage localisation		Damage feature extraction		Damage assessment	Data handling			
	Sensor movement	Surface coverage	Localisation	Classification	Measured unit	Consider damage type		Data storage - comprehensible	Data storage - reproducible	Data modelling	Visualization
Abdel-Qader et al. (Abdel-Qader et al., 2003, 2006)	-	-	**	* Cracks	-	-	-	-	-	-	*
Adhikari et al. (Adhikari et al., 2014, 2013)	-	-	**	** Cracks / Spalling	**	**	*	-	-	-	**
Arena et al. (Arena et al., 2014)	-	-	**	* Cracks	*	**	-	-	-	-	*
Chen et al. (Chen et al., 2012)	-	-	**	* Corrosion	*	*	-	-	-	-	*
Chen et al. (Chen et al., 2011)	**	*	-	-	-	-	-	-	-	-	-
German et al. (now Paal) (German et al., 2012, 2013)	*	*	**	** Cracks / Spalling	-	-	-	-	-	-	*
Guldur, Hajjar (Guldur & Hajjar, 2014)	*	**	***	** Cracks Volumetric damages	**	*	-	-	-	*	-
Koch et al. (Koch et al., 2014)	-	-	***	** Cracks / Spalling	***	**	-	***	*	*	*
Lee et al. (Lee et al., 2008)	**	**	**	* Cracks	-	-	-	**	*	-	-
Li et al. (Li et al., 2014)	*	*	**	* Cracks	**	**	-	*	**	-	**
Matsumoto (Matsumoto, 2013; Matsumoto et al., 2012)	**	**	**	** Cracks / Delamination	**	**	**	-	-	-	**
Mc. Robbie (McRobbie et al., 2011)	*	**	**	* Damages in general	-	-	-	**	*	*	***
Nishimura et al. (Nishimura et al., 2012)	**	**	-	-	-	-	-	*	*	-	-
Oh et al. (Oh et al., 2008)	**	**	**	* Cracks	*	**	-	-	-	-	-
Paal et al. (prev. German) (Paal et al., 2014)	*	*	***	** Cracks / Spalling	-	**	*	-	-	-	**
Pătrăucean et al. (Pătrăucean et al., 2015)	-	-	-	-	-	-	-	-	-	*	-
Tang et al. (Tang et al., 2007)	-	-	-	-	-	-	-	-	-	*	-
Yu et al. (Yu et al., 2007)	-	-	**	* Cracks	**	**	-	-	-	-	**
Zhang et al. (Zhang et al., 2014b)	-	-	**	* Cracks	-	-	-	-	-	-	-
Zhu, Brilakis (Zhu & Brilakis, 2009)	-	-	**	** Air pockets / Discoloration	-	-	-	-	-	-	-
Zhu et al. (Zhu et al., 2011)	-	-	**	* Cracks	***	**	-	*	*	-	*

(- not included / relevant, * basic automation, ** mediocre automation, *** advanced automation)

2.5 Convolutional Neural Network Image Datasets

Every convolutional neural network needs a dataset to be trained. The size and the architecture of the dataset improves the ability of the model to be effective. CNNs can be specifically built and trained differently to reduce the error in a certain evaluation dataset. To compare model performance, researchers have built benchmark datasets for the progression of a certain subject area in machine learning. CNNs train on given data and are able to compare their performance to other models because of the common training and evaluation dataset, i.e *benchmark.*

Benchmark Datasets

A benchmark dataset refers to a dataset which may be referenced by researchers (LeCun, Cortes, & Burges, 1998). Researchers' models may be trained/tested against the benchmark to see how their model compares to others. Researchers will try to train their models to achieve the lowest levels of error. This type of competition is good since superior models will emerge over time. One of the most famous benchmark datasets is the Modified National Institute for Standards in Technology (MNIST) dataset which involves hand-written numbers. This set consists of 60,000 training images and 10,000 test images (LeCun et al., 1998).

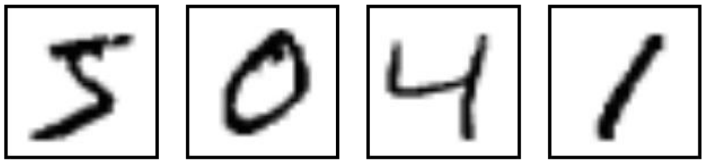


Figure 2.5.1- Sample MNIST Training Dataset (LeCun et al., 1998)

The International Conference on Document Analysis and Recognition (ICDAR) dataset series have been released as a part of an automated reading competition by the ICDAR (Ibrahim, Abbott, & Hussein, 2016). The competition has changed over time and challenges its competitors with new datasets and guidelines to push text-detection boundaries. ICDAR 2003 dataset contained 258 training images and 251 validation images using rectangular bounding boxes around text instances. Eight years later, the ICDAR 2011 dataset was created with 420 images for training and 102 images for testing. Four years after, the

ICDAR 2015 for incidental text instances was developed. Incidental text instances are when text is not the main focus of the image. The most recent example is the COCO-Text dataset for text detection and recognition (Ibrahim et al., 2016). It is derived from Microsoft's common object in context (COCO) dataset. The 2016 version of the dataset contained over 173,000 text annotations in over 63,000 images. The dataset considered legible and illegible text, script of the text, transcriptions, printed, and hand-written text (Veit, Matera, Neumann, Matas, & Belongie, 2016a). This allowed for a robust dataset for advancing state of the art in-text detection and recognition. There are some limitations in the COCO-Text, one of which is having bounding boxes not suitable for most deep-learning systems (Ibrahim et al., 2016).

There are also object-based benchmark datasets like PASCAL Visual Object Classes (Pascal-VOC) challenge. It is a benchmark in visual object category recognition (Everingham, Gool, Williams, & Winn, 2010). Others, like the SUN database for scene recognition in 2010, and a pedestrian detection in 2012 have contributed to the dataset (Lin et al., 2016). ImageNet contains unprecedented numbers of images which has allowed for breakthrough in object classification and detection research (Lin et al., 2016). Then, there is Common Objects in Context or COCO from Microsoft, which was alluded before in the section. This is a newer dataset which explores the nuisances of *non-iconic views* of objects (Lin et al., 2016). *Non-iconic view* means looking at objects which are not front and center, or not the main focus of an image. Another unique feature of the dataset was its segmentation of objects in an image (Figure 2.5.2).

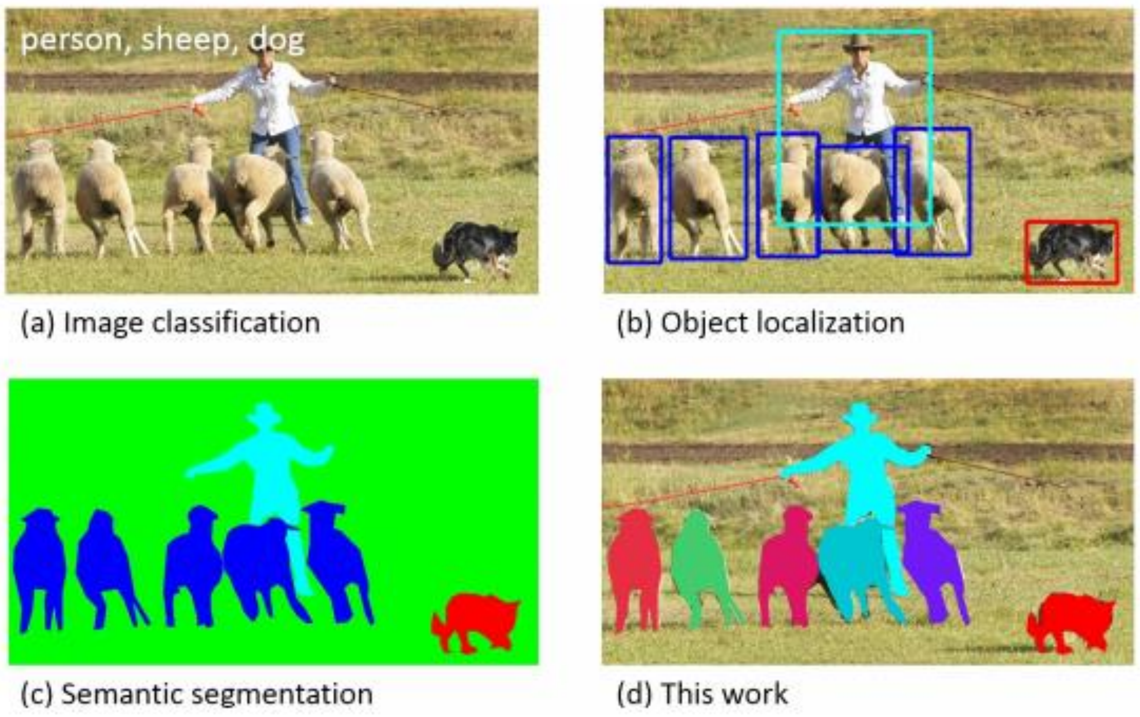


Fig. 1: While previous object recognition datasets have focused on (a) image classification, (b) object bounding box localization or (c) semantic pixel-level segmentation, we focus on (d) segmenting individual object instances. We introduce a large, richly-annotated dataset comprised of images depicting complex everyday scenes of common objects in their natural context.

Figure 2.5.2 - COCO Dataset Enhancements (Lin et al., 2016)

Dataset Labeling Pipeline

Labeling extensive numbers of images is extremely time consuming and costly. Therefore, research on the creation of large datasets have reviewed the optimization of dataset annotation. A unique annotation pipeline was established in the COCO dataset's annotation method. The COCO dataset used Amazon Mechanical Turk to aid in its dataset annotations (Lin et al., 2016). Amazon Mechanical Turk is a way to crowd-source image annotation. The following best-practice steps were identified for the COCO dataset. Figure (2.5.3) helps to illustrate the process.

1. Gather and acquire many images.
2. Label images as containing an object using a hierarchal labeling approach (Deng et al., 2014).
3. Label individual instances
4. Verify instances
5. Segment instances

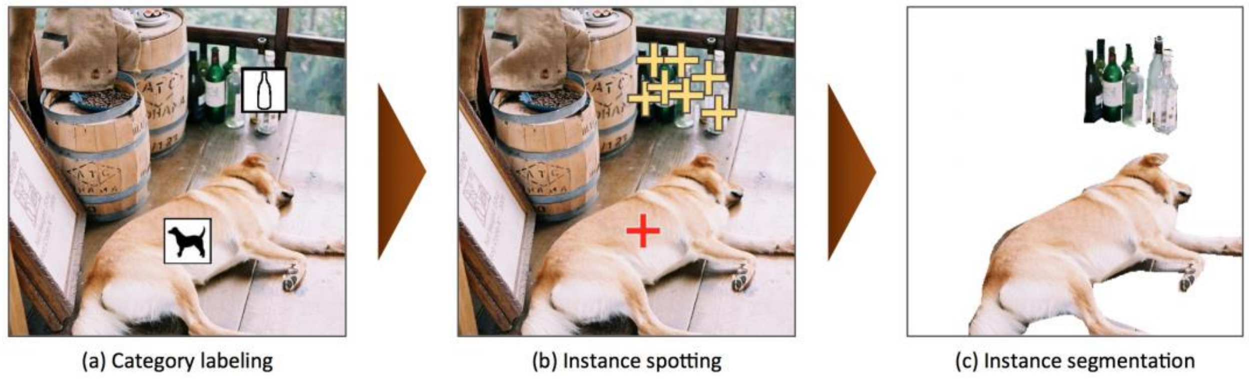


Figure 2.5.3 - Image Annotation Pipeline (Lin et al., 2016)

There are other methods for annotating large sums of images that do not specifically follow the COCO dataset pipeline. Especially since these annotation tools which comprise the COCO pipeline were made specifically for the purpose of their dataset (Lin et al., 2016**)**. The difference lies in the annotation process. The annotation step in the pipeline is crucial in any method of construction for a deep-learning image dataset. Several annotations tools were examined.

After a critical literature review into the annotation tools, it was determined that there are three different functions of annotation tools: image tagging, bounding box, or polygon. LabelMe (Russell, Torralba, Murphy, & Freeman, 2008) is a commonly known annotation tool for segmentation. It can also do bounding boxes and image tagging. There are also plugins for python versions of desktop LabelMe-inspired versions (Js-annotator, VGG image annotator, RectLabel, LabelBox). Some versions only have bounding box abilities, and some only have image segmentation. Additionally, MATLAB has its own version of an annotation tool. This is found by adding on the neural networks plugin. This version allows for images to segmented or defined with bounding boxes in a colorful fashion. A demonstration of this can be viewed in the Figure (2.5.4). Of course, there are annotators which cost money to use, and usually charge by number of images processed or objects annotated (i.e. *Supervisely*).

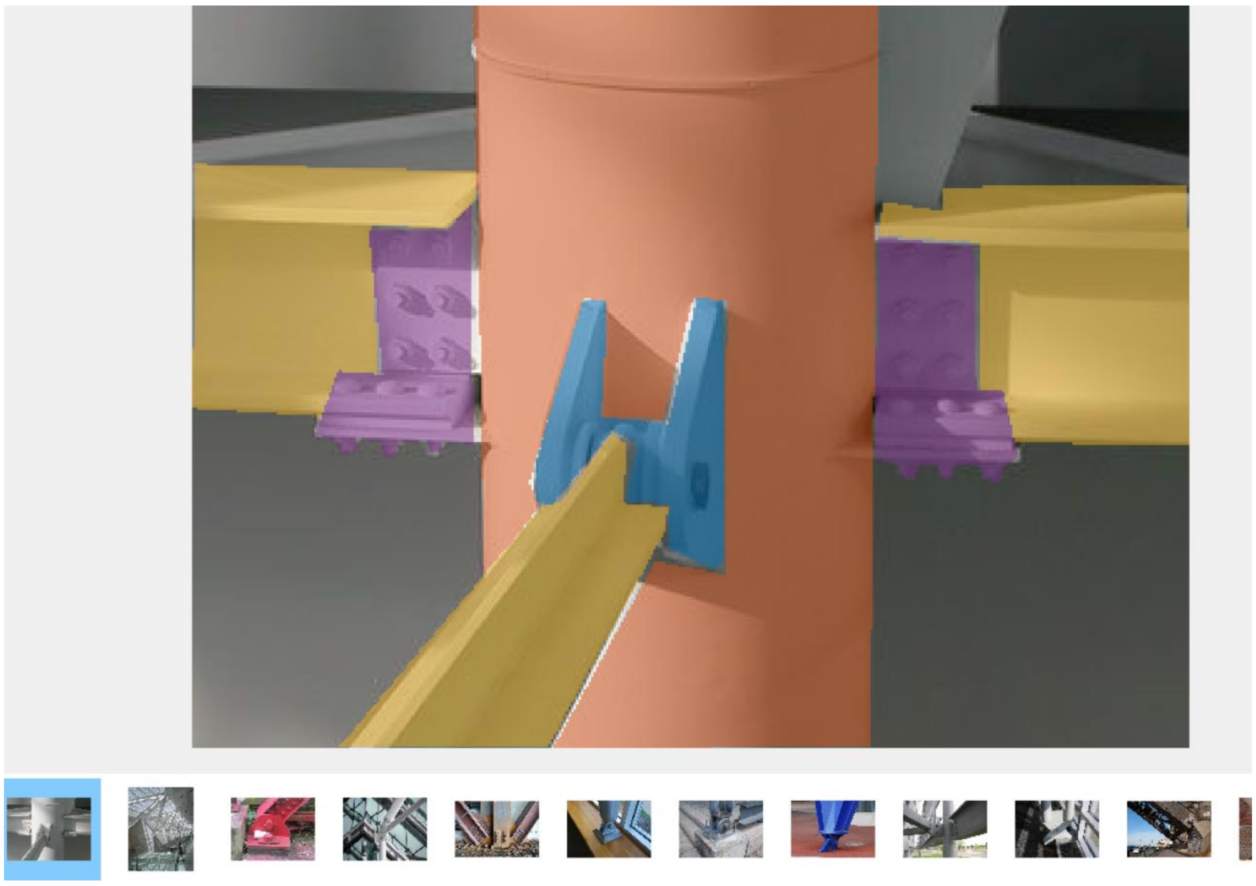


Figure 2.5.4 - MATLAB Image Annotation Tool

Data Augmentation

Images are often augmented to fit the constraints placed on the GPU’s memory during training, or the time it takes for the model to train. For this reason, images may be down-sampled or re-scaled to meet these constraints. Re-scaling should not be confused with re-sizing. Re-scaling an image maintains the aspect ratio, re-sizing an image may not always maintain aspect ratio as in the case. In Figure (3.4.2), the image is re-sized to equal dimensions of height and width, which distorts the image, but would make tracking annotations easier. While re-scaling the images was trickier to implement and track annotations as an image was rotated or mirrored, it was worth not distorting the image.

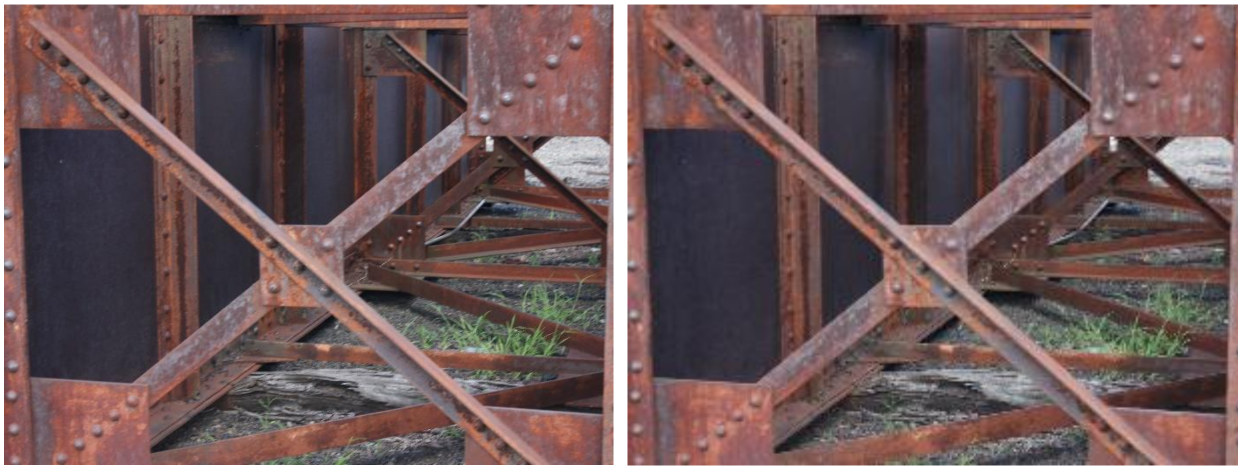


Figure 2.5.4 – Left Image (5184 x 3456) Re-scaled to Right Image (345 x 230)

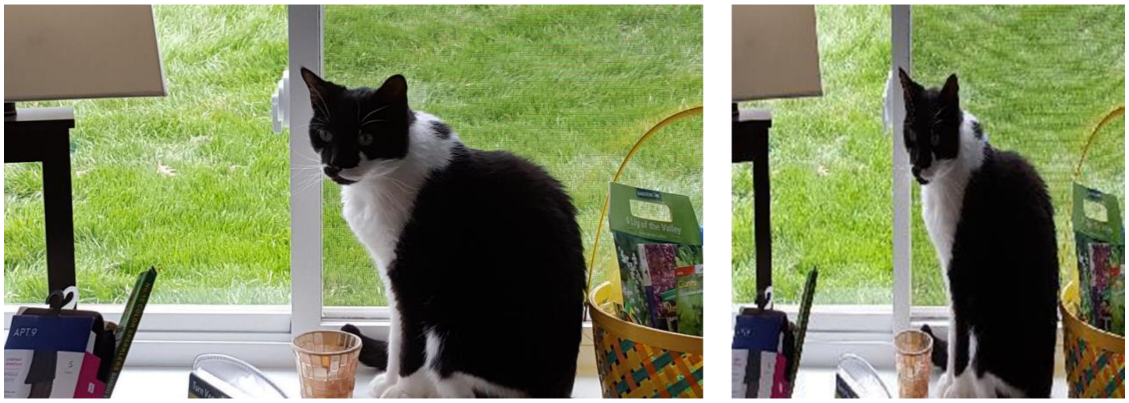


Figure 2.5.5 - Original (right) and Resized to Square (left)

One way to deal with a limited supply of photos is by augmenting the images captured to effectively create more data. The expansion of usable data can be achieved by several methods such as: altering the image through lighting, rotation, blurring, resizing, mirroring, scattering, cropping, zooming-in, etc. While the original image content is the same, the augmentation effectively can turn one data point into many. This is because the computer perceives each adjustment as a completely new image, even though humans recognize it is the “same”. For example, there was a competition to develop a model to classify types of plankton from only a couple hundred photos (Chordia & Verma, 2015). The winning team augmented their data in many ways to achieve accurate results. They used lighting changes, blurring, rotations, mirroring. A research paper at Stanford examined the effects of the augmentation noting 2.2% improved accuracy per epoch with the rotation augmentation and reduction in *overfitting*, which is a dataset balancing issue (Chordia & Verma, 2015). Thus, augmentation has been found to be a common way to boost model performance given a dataset.

Dataset Architecture

Through the literature review process, two considerations were observed in a dataset architecture. The first was how the file is formatted, and the second was balancing the dataset. The file format of the dataset is crucial when it comes to determining the structure of a dataset. It is imperative to output the information in a way which can be readily read by a model. The API of the dataset should be made clear for whomever may be using the dataset in their research. For example, the COCO dataset has an entire section in their website for the file format API (Lin et al., 2016). File formats are not all the same, as the name suggests. Typical file formats in image datasets are XML or JSON (Ankit Gupta, 2017). XML is compact, while JSON can be read by virtually any language and machine. A balanced dataset is important for training and evaluation purposes. Balancing a dataset means having a roughly equal number of object-instances in each class (Ibrahim et al., 2016). This includes having enough images which do not contain any of the desired classes. In autonomous driving, there are severe class imbalances in semantic segmentation since important objects like pedestrians are under-represented unlike the sky and buildings (Siam, Elkerdawy, Jagersand, & Yogamani, 2017). For example, in an a text recognition dataset, COCO-Text-Patch, it was carefully balanced to have half of the images contain text, half of the images not containing text (Ibrahim et al., 2016).

Evolution of Training Models

Three years from the inception of the R-CNN, models have progressed from the R-CNN (Girshick, Donahue, Darrell, & Malik, 2014) to the Fast R-CNN (Girshick, 2015) to the Faster R-CNN (Ren, He, Girshick, & Sun, 2016) and then finally the Mask R-CNN (He, Gkioxari, Dollar, & Girshick, 2017). In summary, the R-CNN was a faster and efficient way to detect objects in images than previously. As the two predecessors suggest, Fast and Faster R-CNN, made the models run faster. The Mask R-CNN extends the Faster R-CNN by adding a branch for predicting an object mask in parallel with the existing branch for bounding box recognition (He et al., 2017). It uses object detection and semantic segmentation to achieve instance segmentation, and combines Faster R-CNN with Fully Convoluted Networks (He et al., 2017). Furthermore, Mask R-CNN uses RoI align, which results in no loss in data information during RoI pooling processes.

2.6 Evaluating Model Performance

The models in this paper were evaluated using common methods found in numerous CNN research papers for bounding box predictions (Girshick et al., 2014; Hu, Gu, Zhang, Dai, & Wei, 2017; Ren et al., 2016; Siam et al., 2017). Common ways that convolutional neural networks are evaluated is through mean average precision (mAP), precision-recall curves, accuracy, and f1-scores. Test thresholds must be determined before doing calculating any of these calculations. These test thresholds are the prediction confidence threshold (λ) and the intersection over union threshold (IOU). The (λ) is how confident the model is that it sees an object it has been trained to look for in an image. If the model is at or above (λ), then the model will record the prediction, otherwise, it will suppress the prediction. The IOU is the threshold that the prediction, i.e. bounding box, was matched to the *ground truth* bounding box (Figure 2.6.1). The *ground truth* bounding box is drawn around objects in the evaluation dataset as the ideal prediction. The better the prediction, the higher the IOU percentage. The way the IOU is calculated is by dividing the area of the overlap by the area of the union (Figure 2.6.1).

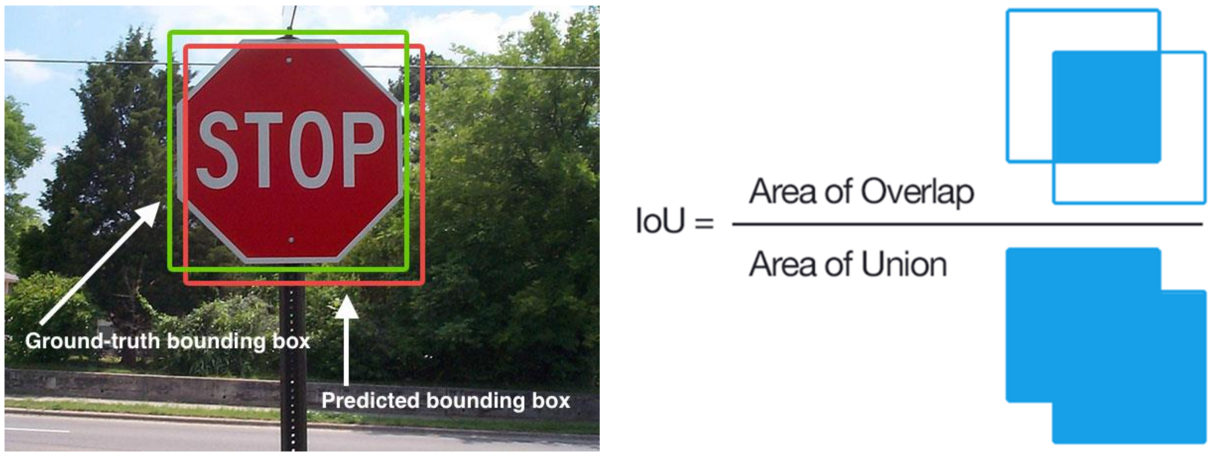


Figure 2.6.1 - Intersection Over Union Example (Rosebrock, 2016)

Should the model have a prediction which is the correct class, and meets the IOU threshold, then that prediction is considered a *true positive* (TP). *False positives* (FP)occur when a prediction does not meet the defined IOU threshold. FP may become TP if the IOU threshold is lowered, however if the prediction is not intersecting any area of the ground truth, then it will always be a FP. *False negatives* (FN) occur when a model fails to predict an object in an image. Among other reasons, FN may be a result from a confidence threshold (λ) which is too high, if the model is overfit to a dataset, or if the model needs more training (more images, more iterations, etc.). For example, if the (λ) is too high it may not be able to show a correct prediction for an object that it sees in an image.

By using the output of all the FP and TP, one can calculate the precision-recall curves, mAP, accuracy, and f1-scores. The following are calculations for precision, recall, and f1-scores.

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

$$Accuracy = \frac{TP}{TP + FN + FP}$$

$$F1 = 2\frac{Precision \times Recall}{Precision + Recall}$$

The precision recall curve is generated by calculating the precision and recall as it runs its prediction on each image in the evaluation dataset. Each class has its own precision-recall curve. When a prediction is made, the confidence level is stored and ranked. The highest confidence level becomes first on the list of predictions across the entire evaluation dataset. When it comes to graphing the curve, most PR curves start at 1.0 precision, since the model is most confident of that prediction (Figure 2.6.2). As the data is added to the graph from the ranked confidence list, the curve may begin to drop in precision since there is a higher likelihood of error from a lower confidence. The area underneath the precision-recall curve is approximated as the average precision for the object class (Figure 2.6.2). The mean of all the class' precision-recall areas under the curve is what comprises the mean average precision (mAP) score.

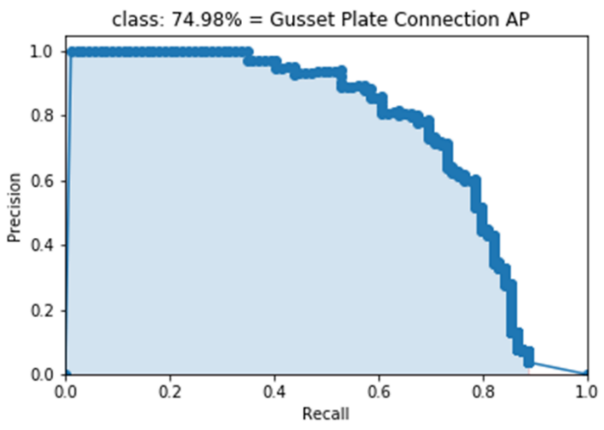


Figure 2.6.2 - Precision Recall Curve

Accuracy measures a model's ability to be close to the true value. In the case of object detection using bounding boxes, accuracy is measured by the number of correct predictions over the number of times the model mis-identifies an object plus the number of actual object instances in the evaluation dataset. The f1-scores were the method of choice to compare models. This is because f1-scores considers both precision and accuracy (Cordts et al., 2016).

CHAPTER 3: METHODOLOGY

3.1 Section Overview

The process of gathering training data and running a model in real-time is explained in the following sections. The process was started by collecting bridge detail images from engineering firms and taking photos of specific bridge details at Norfolk Southern's railway yard. Before collecting the photos, it was important to understand which types of structural details (classes) that were going to be targeted. Once the data was collected, each of the selected bridge details was annotated. The annotation process began with images taken from Norfolk Southern with only three structural detail classes *cover plate terminations*, *gusset plate connections*, and *out of plane stiffeners*. After gathering more photos from AECOM, and many more from Clark Nexsen, a fourth structural detail class was added, *bearings*. The images were then split into training sets, and evaluation sets. Upon completion, the photos were re-scaled, to smaller dimensions, so that the super-computer would not run out of memory during training. Then, the images had the option to be augmented by mirroring and/or rotating. At this point the images and their corresponding bounding box detail files (CSV files) were ready to be converted into tf.records.[2] These special records, along with the training images, are what was referenced during the model training process. The CNN model used was the Single Shot Detection (SSD) v2 with COCO weights (Liu et al., 2016). Figure (3.1.1) is a visual representation of the dataset construction.

[2] This is a special file-type to be read by the TensorFlow API upon training.

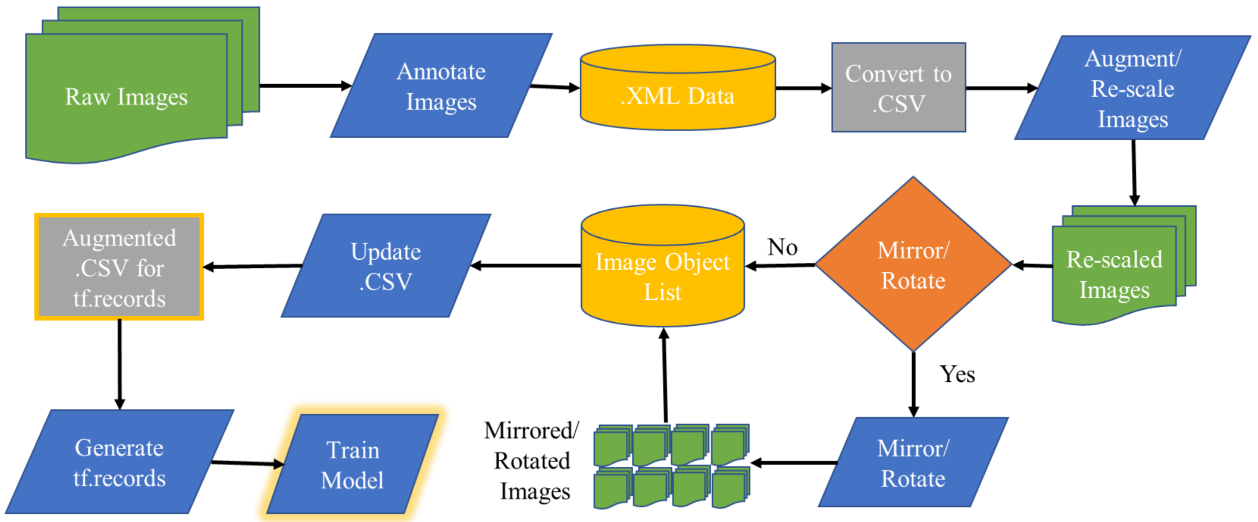


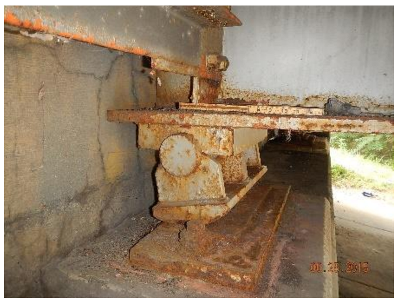
Figure 3.1.1 - Workflow Model for the Dataset Construction

The raw data in the form of images, were gathered. These images were annotated through the use of bounding boxes. The images had corresponding XML data, containing the bounding box information. The XML data was converted to CSV files, which is the format for the tf.records. The images were re-scaled to meet the memory demands of the super-computers. The images were mirrored, rotated, or left as is. Based upon the re-scaling and augmentation, the bounding box dimensions were adjusted accordingly. These alterations were organized by creating a list of image objects. The CSV file was updated based upon the addition of data from augmentation and the re-scaled dimensions. The CSV file was converted into a tf.record, and training could begin.

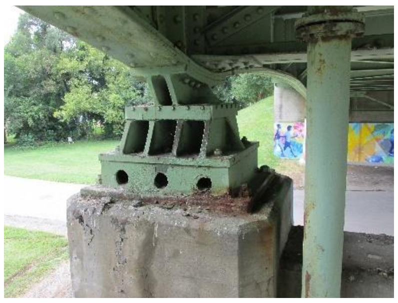
3.2 Data Collection Methods

The goal of the research was to build a dataset which could be used by UAS to identify risk-prone regions in a bridge. The collection of the bridge images was an essential part of creating the dataset. The best way was to build a dataset of images for drone inspection was by using a drone, camera, bucket truck etc. and capture the photos. This would allow one to capture and collect as many details, regions, and angles as desired. The next best option was to out-source the data. This paper out-sourced the data collection to the Clark Nexsen, Virginia Department of Transportation (VDOT), AECOM, and Michael Baker International, all of whom conducted bridge inspections and had access to photos. The photos were collected if the image included one of the four structural detail classes (*bearing, cover plate termination, gusset plate connection or out of plane stiffener*). These structural detail classes were chosen based upon the object instance quantity, and the relevance to bridge inspections.

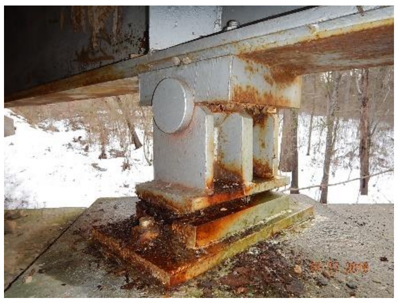
Bearings are a bridge component that transfers loads and movements of the bridge deck to the substructure and foundation (“Bridge Bearings,” 2018). A few example bearings are provided in Figure (3.2.1). These structural details are inspected for rusting and corrosion, or being rocked outside the design tolerance (ODOT, 1973). Being outside the design tolerance may mean that the bearings are excessively tilted or tilted the wrong direction. Image “A” in Figure (3.2.1) shows how the bearing may become tilted. Bearings may also experience fatigue cracking, in which case the inspector would rate the structural detail as critical.



A



B



C

Figure 3.2.1 - Sample Bearings from Collected Images

Cover plate terminations (Figure 3.2.2), were chosen because they are in the category of fatigue-prone details (Table 2.1.2). A cover plate termination is a result from building up the bottom flange of the bridge beam with a welded steel plate. Some examples of *cover plate terminations* from the gathered data are provided in Figures (3.2.3 – 3.2.5). The reason structural engineers have done this was to support the larger moments experienced in the middle of a bridge span. Inspectors must review *cover plate terminations* on the bridge as they are susceptible to fatigue-cracking as shown in Figure (3.2.6).

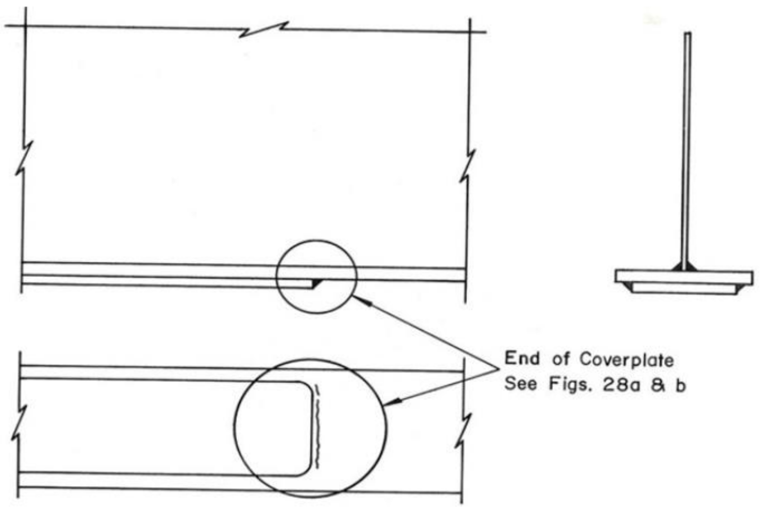


Figure 3.2.2 - Cover Plate Termination (ODOT, 1973)

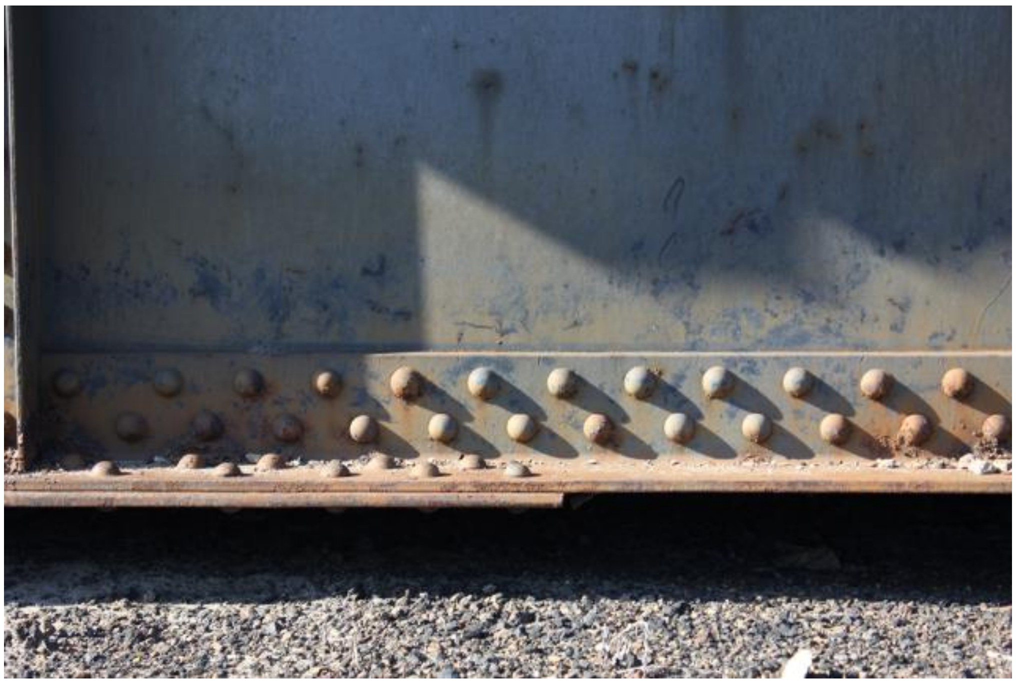


Figure 3.2.3 - Cover Plate Connection from Collected Data

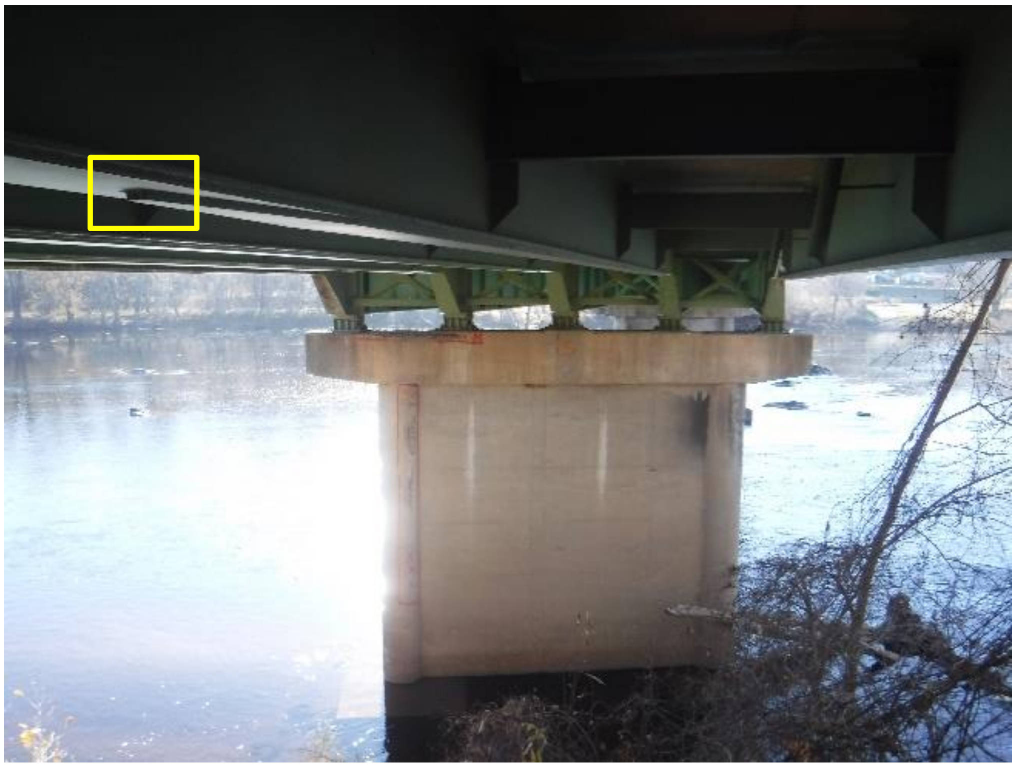


Figure 3.2.4 - Cover Plate Connection from Collected Data

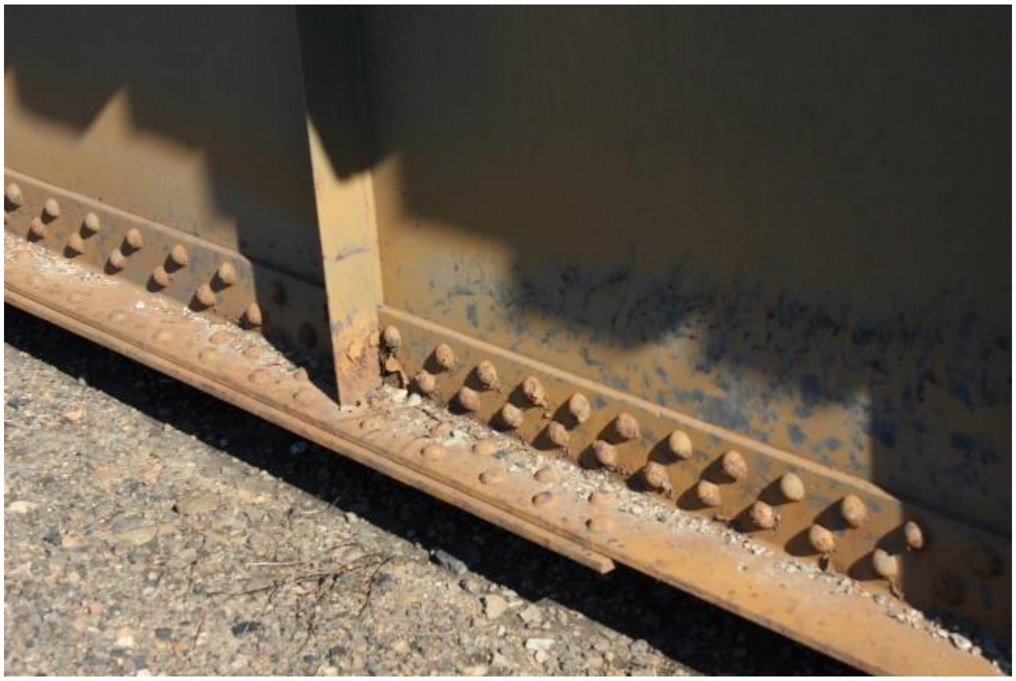


Figure 3.2.5 - Cover Plate Connection from Collected Data

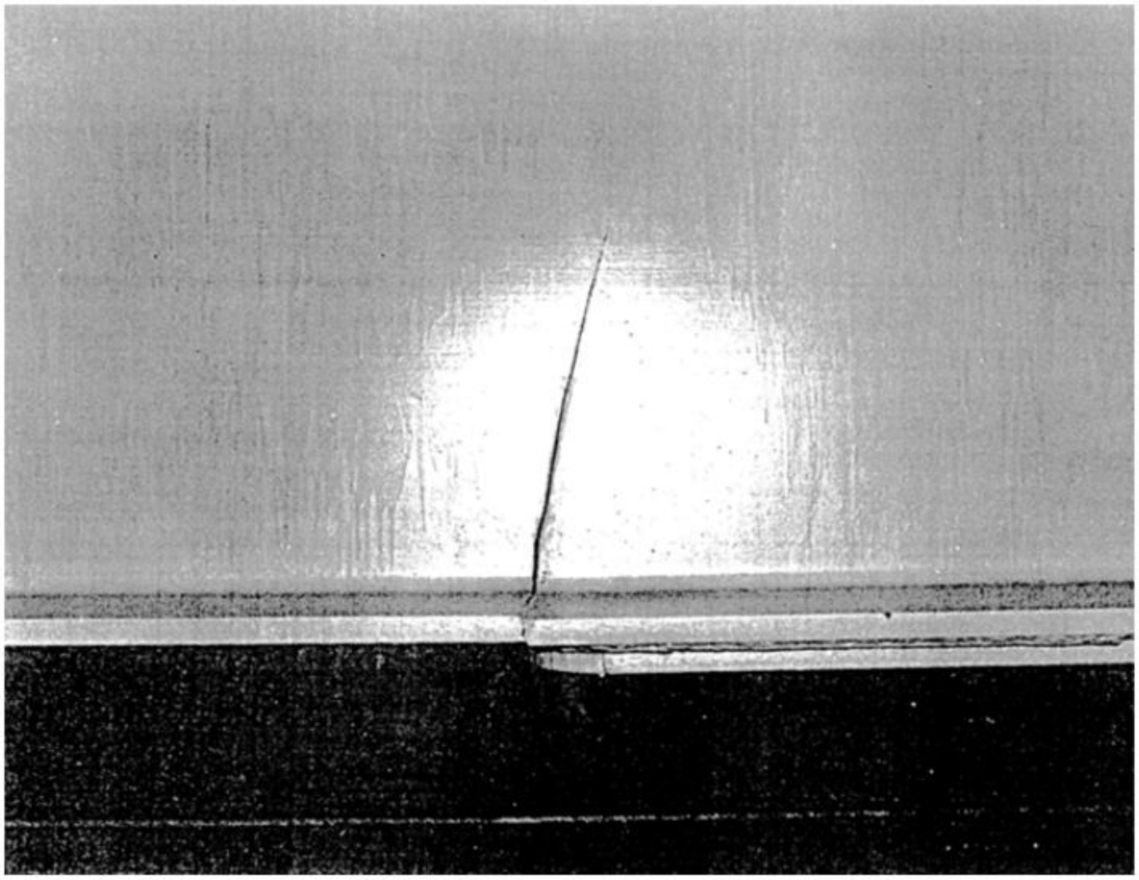


Figure 3.2.6 - Fatigue Cracking in Cover Plate Terminations (ODOT, 1973)

Gusset plate connections are common structural detail in steel bridges (ODOT, 1973). They are found on many bridges including: through trusses, deck trusses, suspension bridges, tied arch bridges, lift bridges etc.) (Figure 3.2.7).

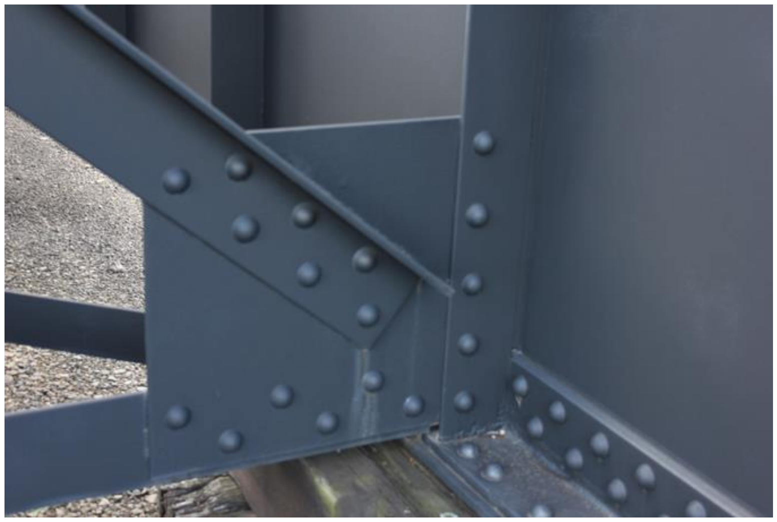


Figure 3.2.7 - Sample Gusset Plate Connection from Collected Data

Gusset plates are designed to transfer tensile and compression axial loads. Inspectors look for section loss, cracking (Figure 3.2.8), and bowing in gusset plates (Figure 3.2.9), since these factors will limit the load-path transfer. Failure to identify deterioration on gusset plate defects could result in costly bridge shut-downs or catastrophic bridge collapses. For example, the I-90 bridge over the Grand River in Ohio had to be shut down for repairs due to a gusset plate fracture in 1996 (Bagnard, 2016).

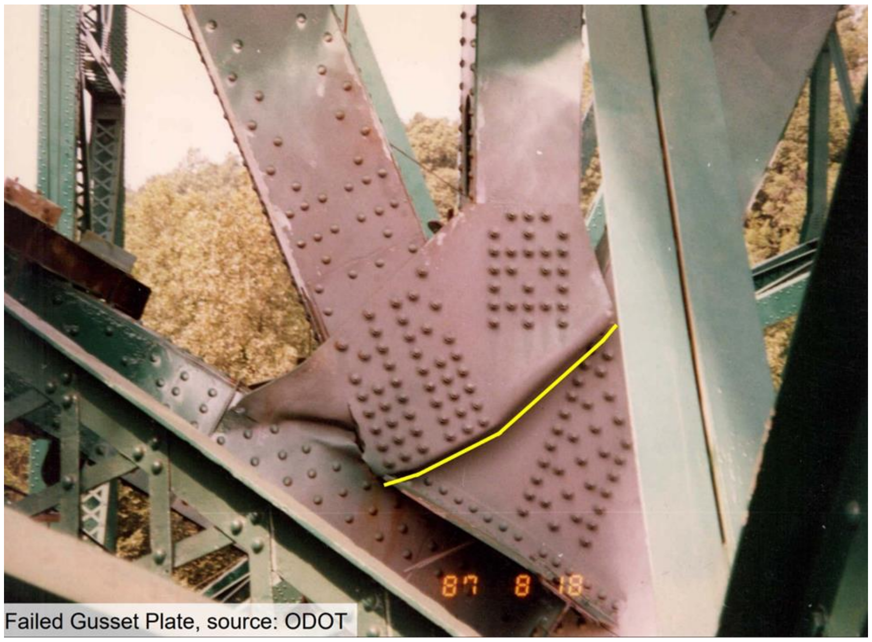


Figure 3.2.8 – Fractured Gusset Plate on I-90 Bridge Over the Grand River, Ohio in 1996 (Bagnard, 2016)

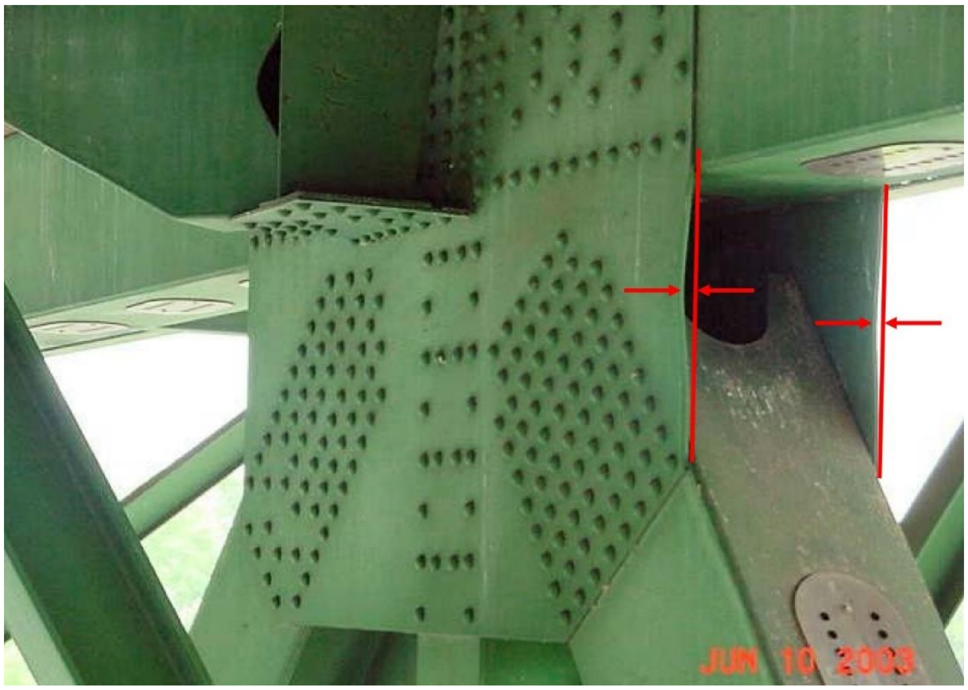


Figure 3.2.9 - Bowing Free-Edge of Gusset Plate (Bagnard, 2016)

Out of plane stiffeners or transverse stiffeners are the fourth structural detail which was collected. *Out of plane stiffeners*, as well as the previously mentioned attachments (*cover plate terminations*, *gusset plate connections*), make up a few items in the list of components welded to a web or flange. Welding introduce internal stresses in the metal which fatigue cracking can form (U.S. Department of Transportation, 2012). In addition to the welds on an *out of plane stiffener* being potentially problematic, old designs of *out of plane stiffeners* are also an issue. For example, the cut-short transverse stiffeners have stress concentrations at the bottom of the stiffener and are prone to develop cracking which propagates from the welded area (Figure 3.2.10). This type of structural detail was based upon assumptions made in the past, modern bridge designs have accommodated for this error by making rigid attachments of the stiffener to the flange, reducing this type of fatigue cracking (Bagnard, 2016).

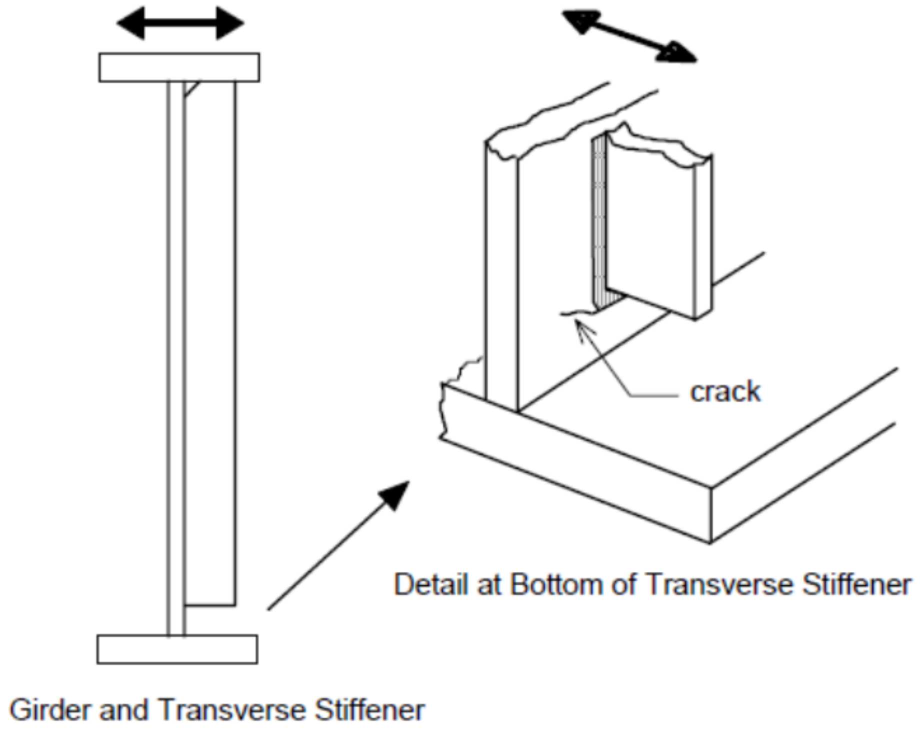

crack
Detail at Bottom of Transverse Stiffener
Girder and Transverse Stiffener

Figure 3.2.10 – Fatigue Cracking on Cut-Short Transverse Stiffener from Out of Plane Movement (U.S. Department of Transportation, 2012)

Images of specific structural bridge detail were collected from Norfolk Southern by means of on-site photographs of railroad bridges which were lying in their railyard. The railroad bridge sections were mainly built-up riveted sections, a style typical of older railway bridge design. The following is a summary table of the data collected from each source (Table 3.2.1).

Table 3.2.1 - Sources and Data

Source	**Type of Media**	**Number**
Clark Nexsen	Image	3200
Norfolk Southern	Image	917
AECOM	Image	250
VDOT	Image	-
Michael Baker International	Video	-

There were limitations from the pictures which were out-sourced (Clark Nexsen, AECOM, and VDOT). The fundamental issue was that the images were collected for a different purpose than training a convolutional neural network. The images were catered to the needs of the inspector, not the needs of training a model. For example, the photos taken by the engineering firms focused primarily on defects. Although the defect was captured, the context of the risk-prone region was sometimes lost. Additionally, regions which did not have an issue, at the time of their inspection, were rarely captured. For the case of this dataset, the non-defective structural detail regions were just as beneficial as the regions with the defects. Effectively, the dataset has generally been limited to deteriorated structural details by the photos out-sourced. In addition to only finding images of deteriorated-focused areas, when searching for structural details in the firm's database, the photos were sometimes hard to find, were time-consuming to extract, or would generate low yields of useable images. Because VDOT's photos were difficult to access and had low yields, they were not used or collected in the data gathering process.

Photos taken at Norfolk Southern Railroad in Roanoke were obtained from an on-site visit of some of their plate girder railroad bridges lying in the railyard. These were decommissioned railroad bridges which were either being re-painted, repaired, or on standby to be used in the event of a bridge section needed to be replaced in the field. Therefore, these photos were well-documented, and did not focus on defects, but rather the structural details themselves. The camera operator had free-reign to adjust the camera's focus and camera angle. For example, the camera focus was occasionally adjusted to introduce blur into the image dataset. By blurring some images, it could increase the robustness of the dataset to be able to handle blurrier images.

Taking on-site photos was a more advantageous method of generating structural detail focused data. The out-sourced data was more difficult to sort through, and often had structural details which were deteriorated, but provided great variety in the data. Michael Baker International provided drone-flight video footage, which would be the most ideal data to be collected. Although it was footage from a drone-flight it did not focus on the selected structural details, which is why it was excluded from the scope of this research.

3.3 Data Annotation

This section references the workflow of the dataset construction (Figure 3.3.1). There were many annotation tools to choose from (Js-annotator, VGG image annotator, RectLabel, LabelBox, LabelMe, Labellmg, Matlab's Annotation tool, and annotation services for cost). These tools were evaluated, and eventually Labellmg was the annotation tool chosen. Each structural detail in the given image was bound by a box and labeled as its corresponding structural detail class. An example window of the program is viewed in Figure (3.3.1), where a *cover plate termination* and an *out of plane stiffener* has been annotated. This annotator tool was chosen because it supports the file format needed (xml), it is python-supported, and it is efficient for drawing and labeling bounding boxes. The file format was chosen to be XML since it could be transferred into a CSV file by using a python script. CSV files were useful for data augmentation and were easy for the TensorFlow API to read.

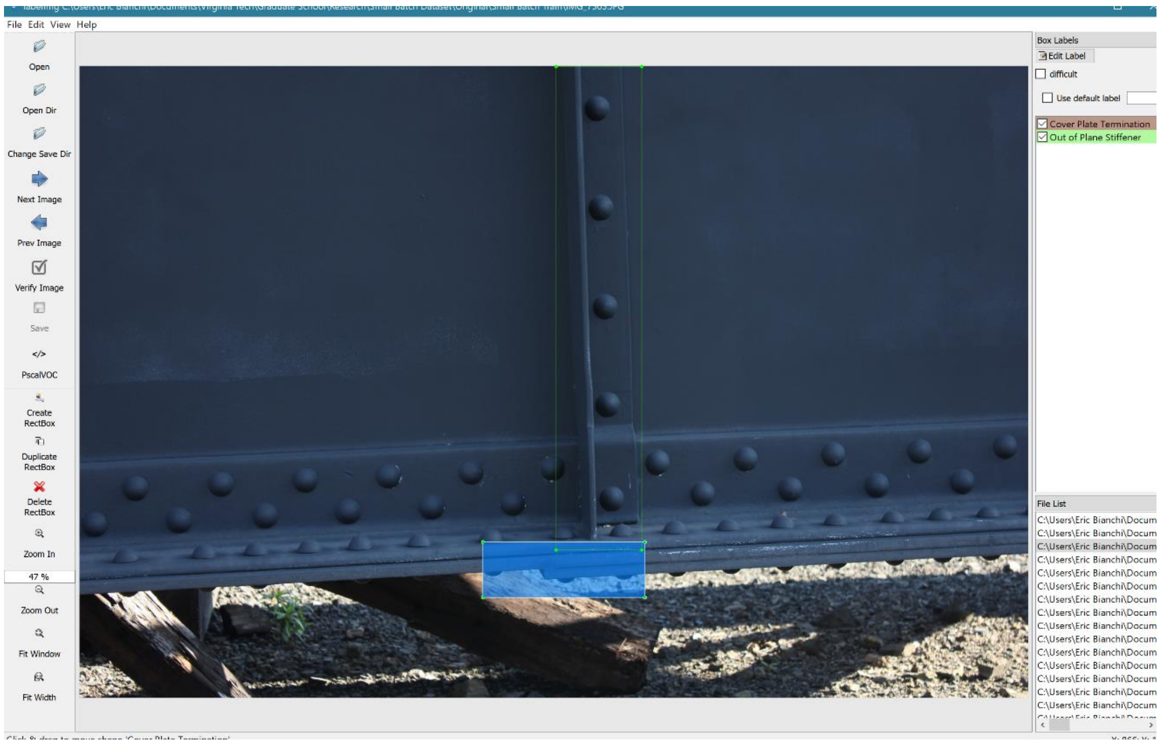


Figure 3.3.1 – Example Window of LabelImg with Cover Plate Termination and Out of Plane Stiffener Annotated.

The annotations were not outsourced due to scope limitations as well as quality control concerns. By having the researcher annotate all the images, it maintained a higher level of quality control from a structural engineer, than from a non-structural engineer. The Norfolk Southern images were annotated first, followed by the outsourced images. The Norfolk Southern images did not contain *bearings*, and were annotated for *cover plate terminations*, *out of plane stiffeners*, and *gusset plate connections*. The data were annotated over a series of months, as images from other sources, like Clark Nexsen, were acquired after Norfolk Southern. Clark Nexsen had images which contained *bearings*, and these images were annotated for all the structural details being trained on. Upon annotation, files were not separated into structural detail folders, because there were often multiple different structural details in each image. However, the photos were separated into steel and concrete bridges. This small adjustment enhanced efficiency and reduced annotation time. After saving each annotated image, there was a corresponding XML file created to be used later in the dataset construction process.

3.4 Data Augmentation Process

Data augmentation began after the images were annotated. By augmenting the data after annotation, it allowed the original data to remain intact as a reference point, should it be used differently in the future. The first step was to generate the CSV files of the bounding box information by running the XML to CSV conversion python script. The number of augmentations of each image was then determined. All images were re-scaled to meet the memory constraints of the GPUs which were used for training[3]. The first run failed because the images were not re-scaled to a size which could be processed by the GPUs. The raw images that were taken at Norfolk Southern and collected from Clark Nexsen were too high of a quality (5184 x 3456). It was found that these images needed to be reduced to dimensions no greater than 400x400 or 500x500. Images were re-scaled to dimensions no greater than 400 through a python script program. A sample of the quality reduction is displayed in Figure (3.4.1). For this dataset, only rotations and mirroring of the original input data were considered. Rotating and mirroring an image in each possible way creates

[3] V100 GPU – CPU: 2 x Intel Xeon Gold 6136 3.0GHz, Cores: 24, Memory: 768GB, 2666MHz

eight augmented images (Figure 3.4.3) and will be referred to as *complete augmentation* through this paper. To wrap up the augmentation process, the CSV files were automatically adjusted to the new dimensions of the re-scaled and augmented images. These newly re-scaled and augmented images, along with their corresponding CSV files are what was used in the training.

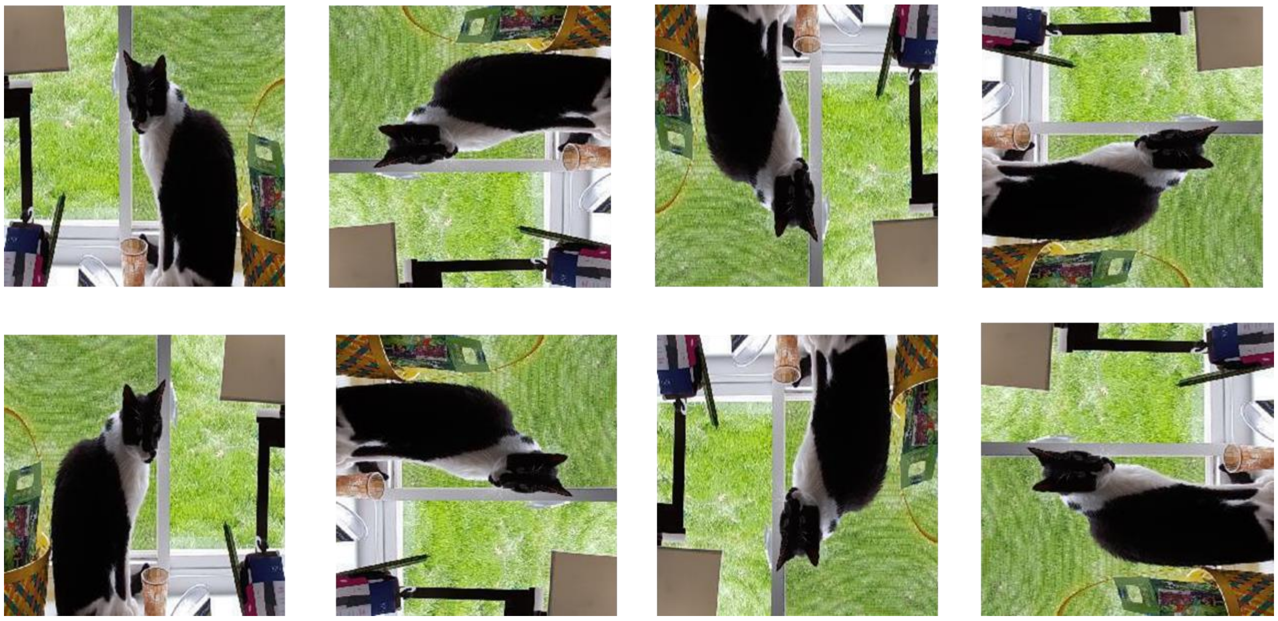


Figure 3.4.1 – Complete Augmentation* of an Image

(From top left to bottom right): Normal, 90-degree rotation, 180, 270, mirror normal, mirror 90, mirror 180, mirror 270

As mentioned in the literature review section, it was important to keep in mind the effects of an unbalanced data set. The COCO-text paper (Ibrahim et al., 2016) found that dataset in-balance affected their model's ability to effectively detect text. Therefore, the augmentation was proposed to maximize the useable data and achieve a more balanced dataset between structural detail classes. An Excel function was developed to augment data based upon the number of structural details in each image. However, the dataset balancing function was never implemented due to scope constraints. Examining model enhancements from a more balanced dataset could be considered in future experimentation.

3.5 Test Variations

There were seven tests which were recorded. The tests were designed to examine overfitting, the effect of augmentation, the effect of selective augmentation, and the effect of step size on model performance. The models were trained on V100 NVIDIA GPUs at Virginia Tech's Advanced Research Center. "Toy model" was the first model which was trained, and it is evaluated against an incomplete evaluation dataset. *Toy model* is more for demonstration and is not used in comparison. Each model had a training dataset and evaluation dataset. Some details were more common than others. The specific quantities of each object instance are presented in Tables (3.5.2 – 3.5.9) for each model, and is summarized in Table (3.5.10, 3.5.11). It is standard practice to separate evaluation ~80-85% training and ~15-20% evaluation, as done in "The Cityscapes Dataset for Semantic Urban Scene Understanding" (Cordts et al., 2016). In this paper, the researchers split their data 2975 training and 500 evaluation. However, this consideration depends on the number of images available to use, and refining this split is not black and white. For COCO-Bridge, it was decided that having more training images was better than having a higher ratio split of evaluation to training data. The COCO-Bridge dataset, at its final build of 4 classes, has approximately a 1:10 ratio of training to evaluation images (Table 3.5.12). In the future it would be better to parcel the evaluation dataset more evenly among structural detail instances as well as increase the number of evaluation images.

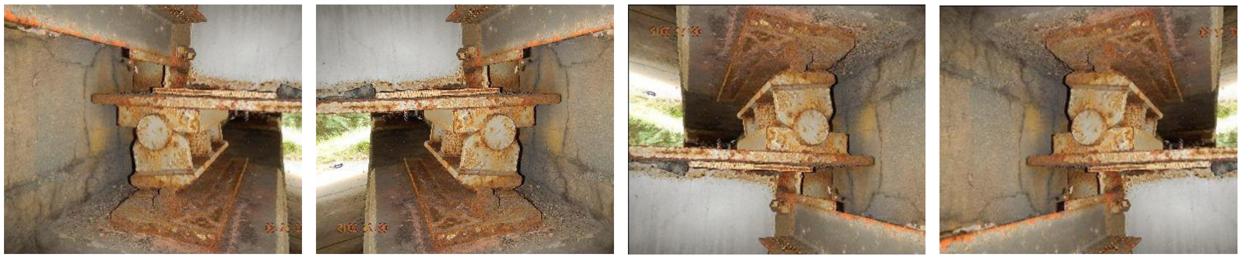
The models and datasets were given a specific nomenclature to abbreviate their title. For example, consider the model Control-3c-5k. The "3" was in regard to the number of "c" classes trained. The "5k" was referring to the number of computational steps (5000). The datasets were ordered alphabetically and use the subscript "T" and "E" to describe that it was a *training set,* or an *evaluation set* respectively.

Overfitting Evaluation

Control-3c-5k uses training set (A_T), and is trained only on images from a single source, Norfolk Southern. These training images, from Norfolk Southern, were predominately of railway bridges. The Control-3c-5k model was evaluated on images from Clark Nexsen and Norfolk Southern (B_E). Additional Data-3c-5k was trained on (B_T), a set of images from Clark Nexsen and Norfolk Southern, and evaluated on evaluation set (B_E). Control-3c-5k and Additional Data-3c-5k are compared against each other to examine if the model overfit. Theoretically, Additional Data-3c-5k, the model with the additional data from Norfolk Southern and Clark Nexsen, would perform only slightly better than the Control-3c-5k, the model trained only on Norfolk Southern images, since the evaluation dataset included a wider variety of images. Additional Data-3c-5k was better suited to handle the wider variety of images which were drawn from both the Norfolk Southern and Clark Nexsen. If the model over-fit, then Control-3c-5k would perform significantly worse than Additional Data-3c-5k. Overfitting occurs when a model performs well on data that is similar to what it trained on but does poorly on data which is very different "looking" than what it was trained on. For example, if model were trained only on *gusset plate connections* from railway bridges, and it only could detect *gusset plate connections* on railway bridges, then the model is overfit. If the same model could detect gusset plate connections on many type of bridges or make correct predictions on types of *gusset plate connections* that it has never seen, then the model would not be overfit.

Effect of Augmentation

Augmented-4c-5k completely augments each image 8 different ways by mirroring and rotating each image (Figure 3.5.1). The annotations follow those augmentation movements, effectively annotating 8 times more images than before. This type of augmentation was referred to as *complete augmentation* throughout the rest of this paper. The motivation behind *complete augmentation* was to increase the size of the dataset, while also introducing randomness. It was hypothesized that model would be able to benefit from the augmentation and better handle any structural detail pattern/placement. The effect of increasing the dataset size as well as introducing randomness was evaluated against Selective Augmentation-4c-5k (selective augmentation), and the control, Control-4c-5k (no augmentation).



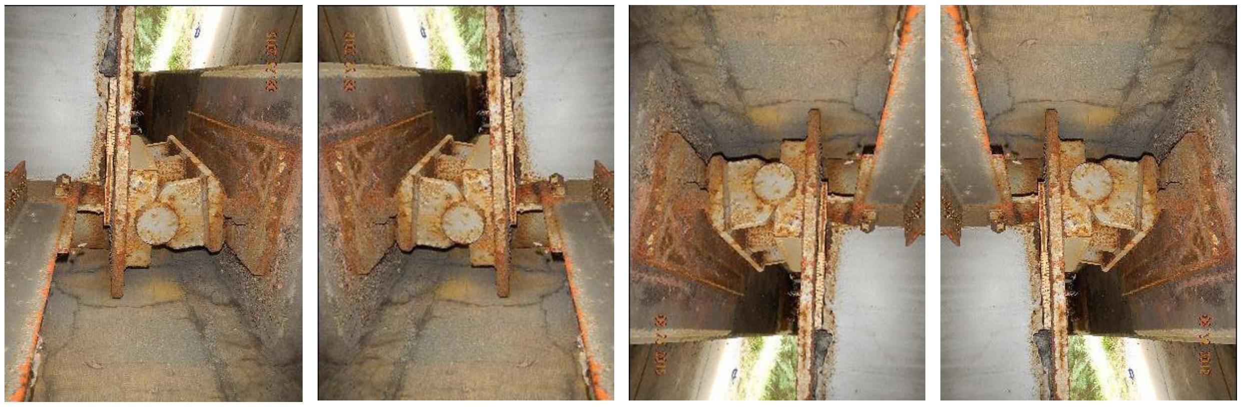


Figure 3.5.1 - Complete Augmentation

Effect of Selective Augmentation

When doing preliminary tests, on datasets unrelated to bridges, it was found that *complete augmentation* of images of vehicles yielded poor results when compared to un-augmented vehicle images. It was hypothesized that *complete augmentation* skewed reality, and confused model more than it helped when testing against the evaluation dataset. In order to increase the number of images, while maintaining realistic geometry, *selective augmentation* was hypothesized to outperform complete augmentation. For *selective augmentation*, the image was only be mirrored across the y-axis (Figure 3.5.2). It only introduces image geometry augmentations which are realistic. In a realistic world, one would not be looking at a bridge upside-down, or even from a 90-degree angle. Training a model with these skewed viewpoints may confuse it more than help it. Therefore, complete augmentation may only be useful for structural details which do not require any specific orientation. This test was compared against, Augmented-4c-5k, Control-4c-5k, and Control-4c-10k.

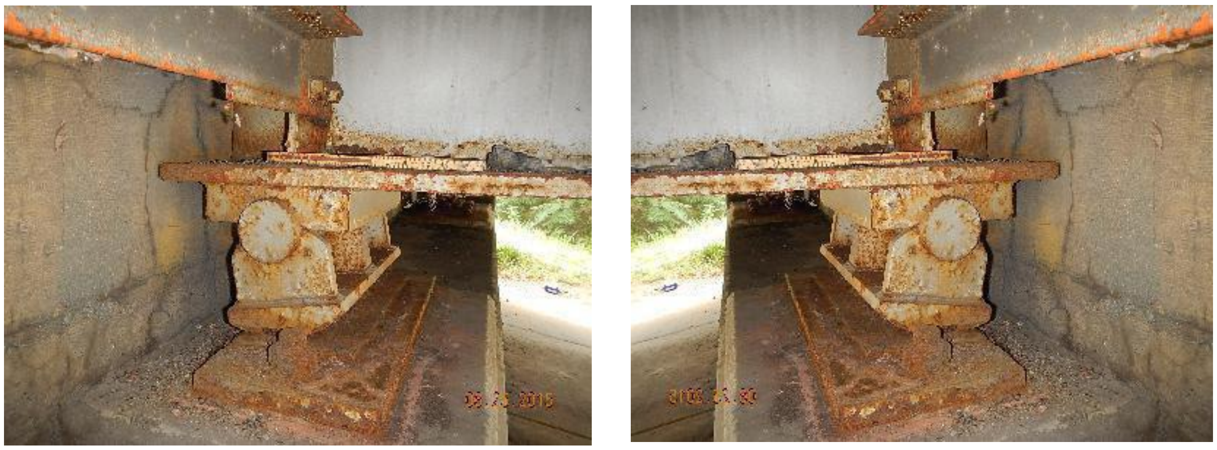


Figure 3.5.2 - Selective Augmentation

Effect of Increasing the Number of Steps

Increasing the number of steps in a model effectively refines the process of learning. Eventually the learning rate, or error loss, plateaus and the number of steps become frivolous. The step number refers to the number of iterations or computational steps taken during the training process and is instantiated in the configuration file. All the models, except the Control-4c-10k were trained with 5,000 steps. The TensorFlow scalar graphs provide a visual representation of the error loss v. the computational steps found in Appendix E.

Table 3.5.1 - Test Set Description

Test Name	*Training Set*	*Evaluation Set*
"Toy" model	A_T	A_E
Control-3c-5k	A_T	B_E
Additional Data-3c-5k	B_T	B_E
Augmented-4c-5k	C_T	C_E
Selective Augmentation-4c-5k	D_T	C_E
Control-4c-5k	E_T	C_E
Control-4c-10k[4]	E_T*	C_E

Table 3.5.2 - Training set 'A_T'

Class	*Number of Object Instances*	*Total Number of Images*	*Data Origin*
Gusset Plate Connection	359	414 (345) x (230)	Norfolk Southern
Out of Plane Stiffener	787		
Cover Plate Termination	195		
Bearing	0		

[4] This test was identical to Control-4c-5k except that the number of step sizes were increased from 5000 to 10000.

Table 3.5.3 - Evaluation set 'A_E'

Class	*Number of Object Instances*	*Total Number of Images*	*Data Origin*
Gusset Plate Connection	22	8 (345) x (230)	Norfolk Southern
Out of Plane Stiffener	12		
Cover Plate Termination	2		
Bearing	0		

Table 3.5.4 - Train set 'B_T'

Class	*Number of Object Instances*	*Total Number of Images*	*Data Origin*
Gusset Plate Connection	580	592 (345) x (230)	Norfolk Southern
Out of Plane Stiffener	1131		
Cover Plate Termination	200		
Bearing	0		

Table 3.5.5 - Evaluation set 'B_E'

Class	*Number of Object Instances*	*Total Number of Images*	*Data Origin*
Gusset Plate Connection	75	36 (345) x (230)	Norfolk Southern, Clark Nexsen
Out of Plane Stiffener	80		
Cover Plate Termination	12		
Bearing	0		

Table 3.5.6 - Train set 'C_T'

Class	*Number of Object Instances*	*Total Number of Images (Augmented)*	*Data Origin*
Gusset Plate Connection	4504	5752 (230-400) x (230-400)	Norfolk Southern, Clark Nexsen
Out of Plane Stiffener	8816		
Cover Plate Termination	1576		
Bearing	3800		

Table 3.5.7 - Evaluation set 'C_E'

Class	*Number of Object Instances*	*Total Number of Images*	*Data Origin*
Gusset Plate Connection	87	55 (273-400) x (230-345)	Norfolk Southern, Clark Nexsen
Out of Plane Stiffener	98		
Cover Plate Termination	14		
Bearing	20		

Table 3.5.8 - Train set 'D_T'

Class	*Number of Object Instances*	*Total Number of Images (Select Aug.)*	*Data Origin*
Gusset Plate Connection	1126	1438 (230-400) x (230-400)	Norfolk Southern, Clark Nexsen
Out of Plane Stiffener	2204		
Cover Plate Termination	394		
Bearing	950		

Table 3.5.9 - Train set 'E_T'

Class	*Number of Object Instances*	*Total Number of Images*	*Data Origin*
Gusset Plate Connection	563	719 (256-400) x (230-400)	Norfolk Southern, Clark Nexsen
Out of Plane Stiffener	1102		
Cover Plate Termination	197		
Bearing	475		

Table 3.5.10 – Training: Object Instance Count

Model	*Gusset Plate*	*Out of Plane Stiffener*	*Cover Plate Termination*	*Bearing*
A_T	359	787	195	0
B_T	580	1131	200	0
C_T	4504	8816	1576	3800
D_T	1126	2204	394	950
E_T	563	1102	197	475

Table 3.5.11 – Evaluation: Object Instance Count

Model	*Gusset Plate*	*Out of Plane Stiffener*	*Cover Plate Termination*	*Bearing*
A_E	22	12	2	0
B_E	75	80	12	0
C_E[5]	89	116	14	27

Table 3.5.12 – Ratio of Evaluation to Training Set

Model	*Gusset Plate*	*Out of Plane Stiffener*	*Cover Plate Termination*	*Bearing*
A_E	6.1%	1.5%	1.0%	-
B_E	12.9%	7.1%	6.0%	-
C_E	15.8%	10.5%	7.1%	5.7%

3.6 Training the Model

Optimization of the CNN models were not considered in the training process. The goal of the research was not to determine the best neural network model for the dataset, however the COCO-trained Single Shot Detection (SSD) model was a viable option (Liu et al., 2016). This model was found through an open-source community github account with a variety of COCO-weighted models[6]. The COCO-weighted SSD model was chosen based on convenience, and balance between speed and mAP performance. Speed and performance would need to be considered should a CNN model be used for UAS bridge inspection. The

[5] The full set of evaluation images are provided in Appendix C

[6] https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/detection_model_zoo.md

specific version chosen was the ssd_inception_v2_coco. It is likely that there are models in literature which would out-perform the ssd_inception_v2_coco, however it was outside the scope of the project to investigate other options.

In terms of training, the configuration file for the model had certain parameters such as *number of steps*, *batch size*, and *learning rate*, among others, which could be altered to potentially improve performance. Since model performance was not the focus of the research, these considerations were lightly considered. The number of computational iterations or *number of steps* were the only parameter in the training configuration file to be altered. There were 5,000 *steps* for all the tests except for the Control-4c-10k, which had 10,000 steps. The *batch size* was kept at 24, which was the default number in the configuration file. The *batch size* controls the number of training examples in each forward and backward pass. Increasing the *batch size* increases the amount of memory required during training. This limitation was another reason that the batch was not altered.

For each test, a zipped folder containing the tf.records, images, and necessary supporting files were uploaded into Virginia Tech's Advanced research computers. The files were extracted, and a shell script file was used to run the training file. After the model successfully was trained, the checkpoint files were zipped back up and downloaded back to a local computer for post-processing evaluation. The code for initializing test set-up as well as training the model was stored in a github account[7]. The models were developed in python, using TensorFlow's convolutional neural network API. More information on the code architecture may be found in Appendix A.

[7] https://github.com/beric7/COCO-Bridge

3.7 Testing

A proof of concept of how the model may be used in UAS practice was developed and is shown in Figure (3.8.1). The idea is that a drone would have a camera on-board and relay the video feed back to a control station or first-person view (FPV) goggles that an inspector was wearing. The model predictions would be super-imposed on the video feed to cue the inspector to the locations which needed addressing. An FPV camera with a transmitter antenna was used to capture footage. A receiver, which was plugged into a laptop was used to receive the video feed. A python script was written to impose the model predictions over the video feed. The model predictions were for a vehicle dataset which was developed before training on the structural detail dataset. A list of materials has been consolidated in Table (3.8.1). The intent was to apply the structural model onto a drone for an in-field test, however, there was not enough time in the schedule for an in-field test.

Table 3.7.1 - Materials List for Proof of Concept

Equipment	Power
Camera: FPV Camera System (3 in 1) Transmitter, antenna, camera	**Battery**: Noiposi 3.7 V, 850 mAh: Nano lithium polymer battery
Receiver: Eachine ROTGO1 5.8G 150CH	**Computer**

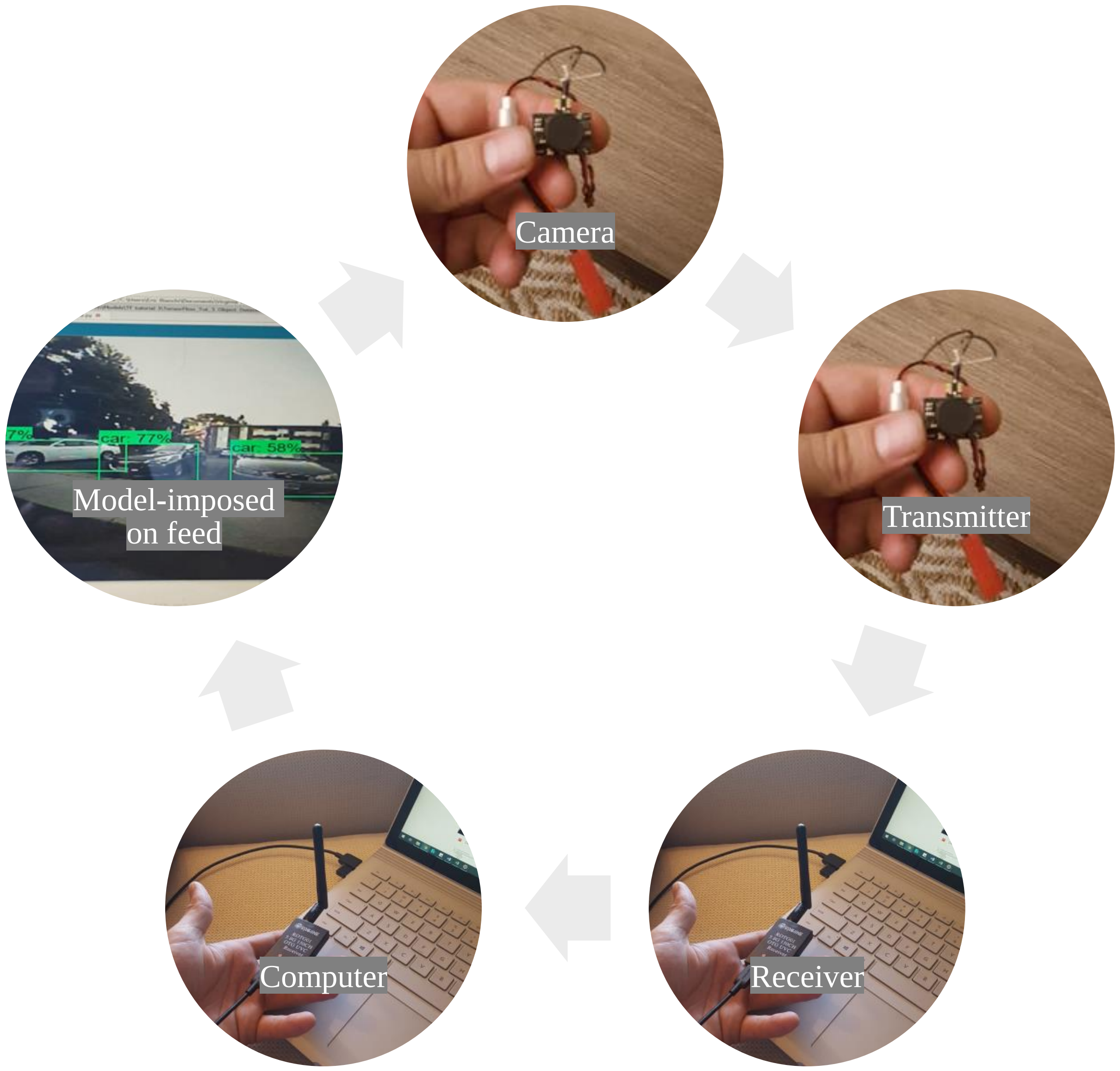


Figure 3.7.1 - Proof of Concept Test

CHAPTER 4: EXPERIMENTAL RESULTS

4.1 Commentary and Reasoning

The mean average precision (mAP) for evaluating model is shown in section 4.2, and the raw data is found in Appendix B. Three mAP graphs were made for each IOU level tested (35%, 50%, 75%). The IOU is commonly found to be 50% in literature (Girshick, 2015; He et al., 2017; Veit, Matera, Neumann, Matas, & Belongie, 2016b), and in some cases 75% (Hu et al., 2017). However, after running some preliminary tests, and viewing the results, it was decided that an IOU of 35% would also be useful for evaluation. For example, the obscurity and objectiveness of the bounding box size for structural details such as *cover plate terminations* Figure (4.1.1) or the thickness of an *out of plane stiffener* Figure (4.1.2), skew results of otherwise precise matches. Because these details are long and skinny, it sometimes is difficult for the model to achieve an IOU of 50%. In the Figure (4.1.1), the blue bounding boxes are the ground truth, the green are true positive predictions that met the IOU threshold, and the red bounding boxes are predictions which did not meet the IOU threshold. Considering the Control-4c-5k model IOU of 50% and 35%: each class's precision benefitted from a lower limit on the IOU. *Cover plate terminations* and *out of plane stiffeners* increased by the most precision 23% and 21% respectively. After viewing the results, and in the use-case for UAS bridge inspections, an IOU of 35% was a reasonable perspective in determining performance. The 35% IOU threshold was the only threshold tested under the standard 50%.

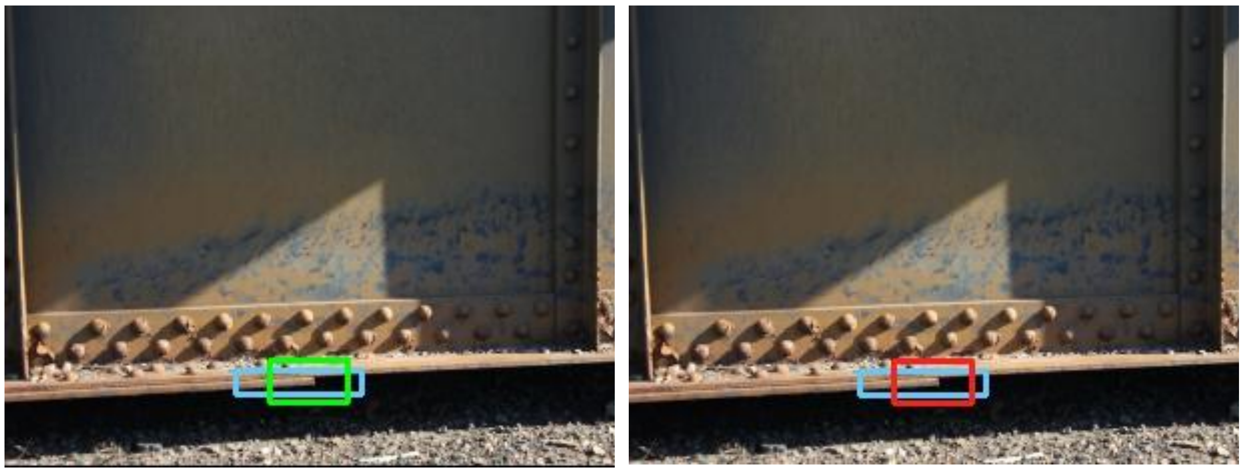


Figure 4.1.1 - Cover Plate Termination Predication for Control-4c-5k

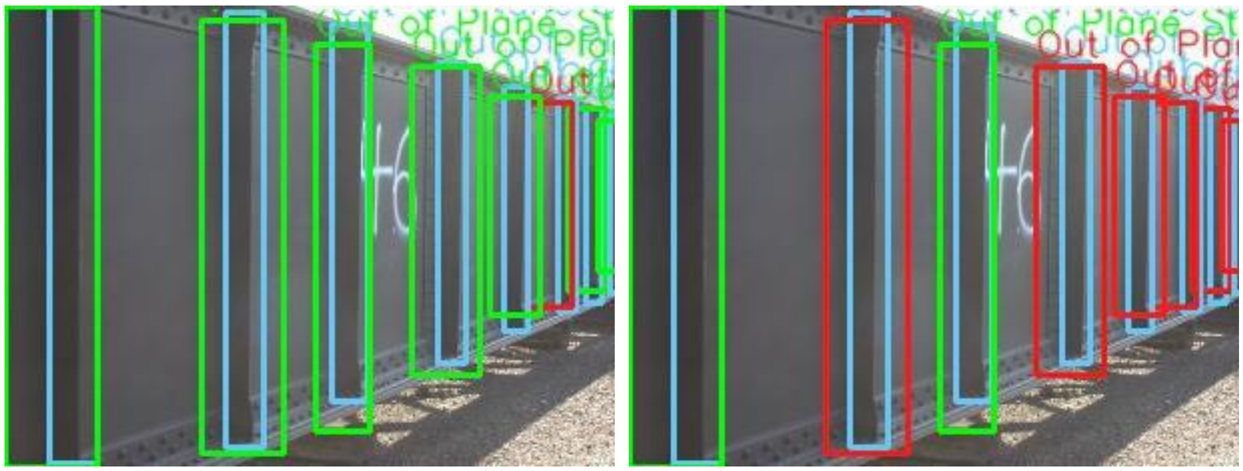


Figure 4.1.2 - "Skinny" Evaluation Criteria for Out of Plane Stiffeners for Control-4c-5k

There were four mAP points on each graph, representing a different threshold (λ). The (λ) is the minimum class confidence limit that the model allows for a prediction to be made on an image. For example, should the model's (λ) be 50%, and it is 49% confident that an image has a bearing, then it would not output that prediction. The four model (λ) chosen were 10%, 25%, 50%, and 75% to give a reasonable curve to determine model performance at varying levels of confidence. It was found that as the minimum (λ) dropped much further than 10%, unreasonable predictions made the model impractical for its use case. An example output of unreasonable predictions is exaggerated by displaying the output from a (λ) of 1% in Figure (4.1.3). In addition to the mAP graphs, average precision for each structural detail and average precision across all classes were graphed to evaluate any enhanced class performance by a dataset and overall precision trends. Should the dataset be used for inspector assistance, an inspector would not want their field of view obstructed by superfluous predictions, even if it did not capture all structural details (accuracy) in the image. The precision by structural detail allowed for an evaluation of the strength of each class, which would otherwise be masked by mAP. The mAP scores were calculated by estimating the area under the precision-recall curves as defined in the literature review section (Figure 2.6.2, Figure 4.3.1). The precision-recall curves did not reach a recall of 1.0, like examples one may find in literature, but that is because the (λ) tested were too high and did not allow the curve to reach a recall of 1.0. However, when the (λ) was lowered to 1% Figure (4.3.2), the curves appear more like what one would expect a precision recall curve to look like.

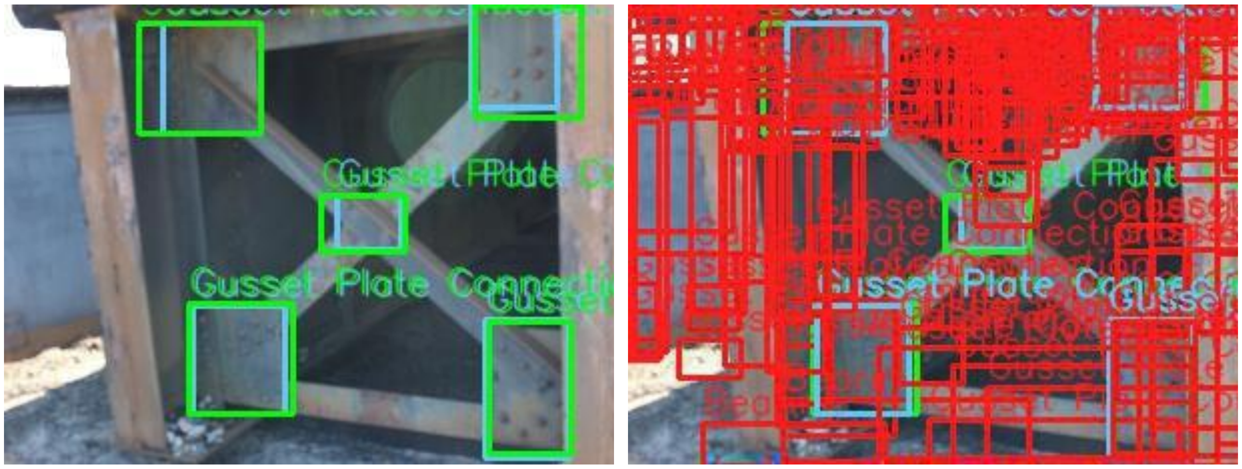


Figure 4.1.3 – Threshold λ=10% (left) and λ=1% (right) for Control-4c-5k

Accuracy was also recorded in similar fashion as the precision. Therefore, accuracy of the model runs for each of the thresholds and IOUs were calculated and graphed. The accuracies were broken down by structural detail as well as accuracy averaged across classes. Because both accuracy and precision are important for an inspector, it was determined that accuracy and precision should be used for performance evaluation. A common way in literature to consider both precision and accuracy is through f1-scores (Singh & Garzon, 2016). These f1-scores were broken down in the same way as the mAP graphs. Three mean f1-scores graphs were made for each IOU threshold (35%, 50%, 75%) to encompass accuracy and precision and determine overall performance.

4.2 Mean Average Precision

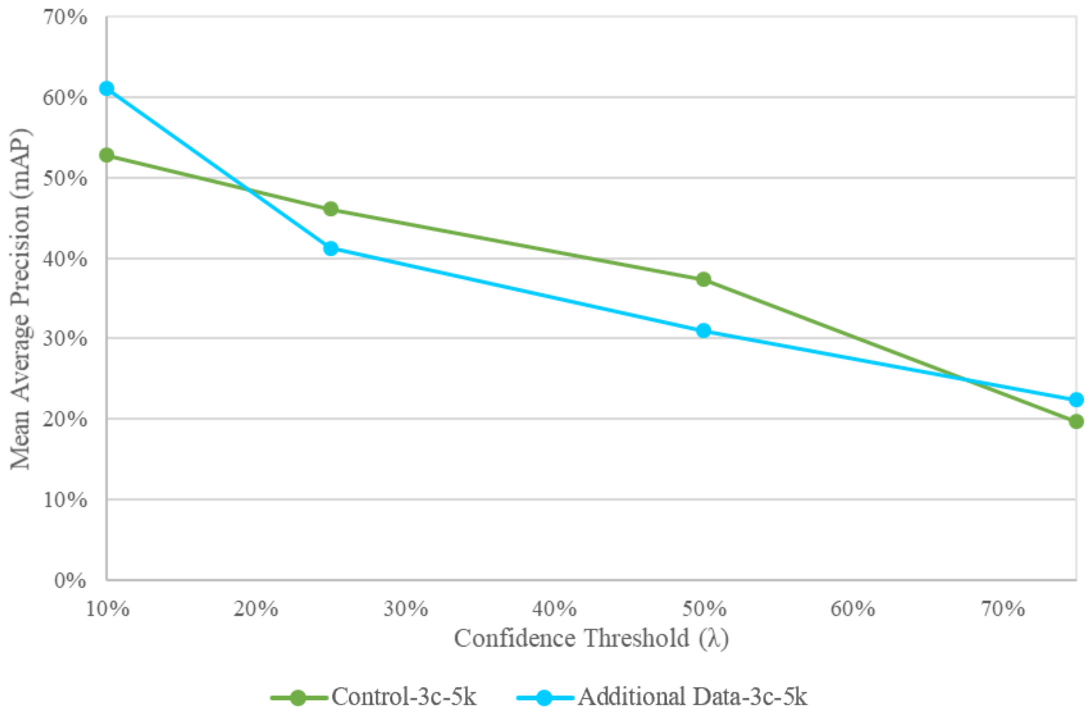


Figure 4.2.1 - IOU 35% Overfitting Evaluation

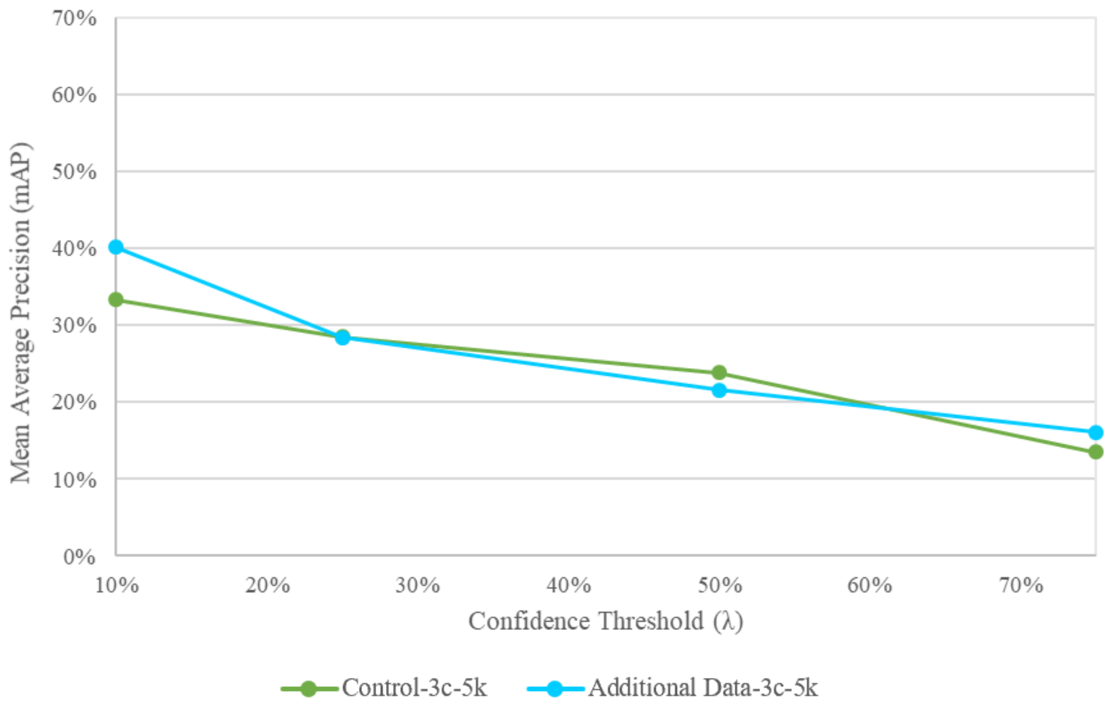


Figure 4.2.2 - IOU 50% Overfitting Evaluation

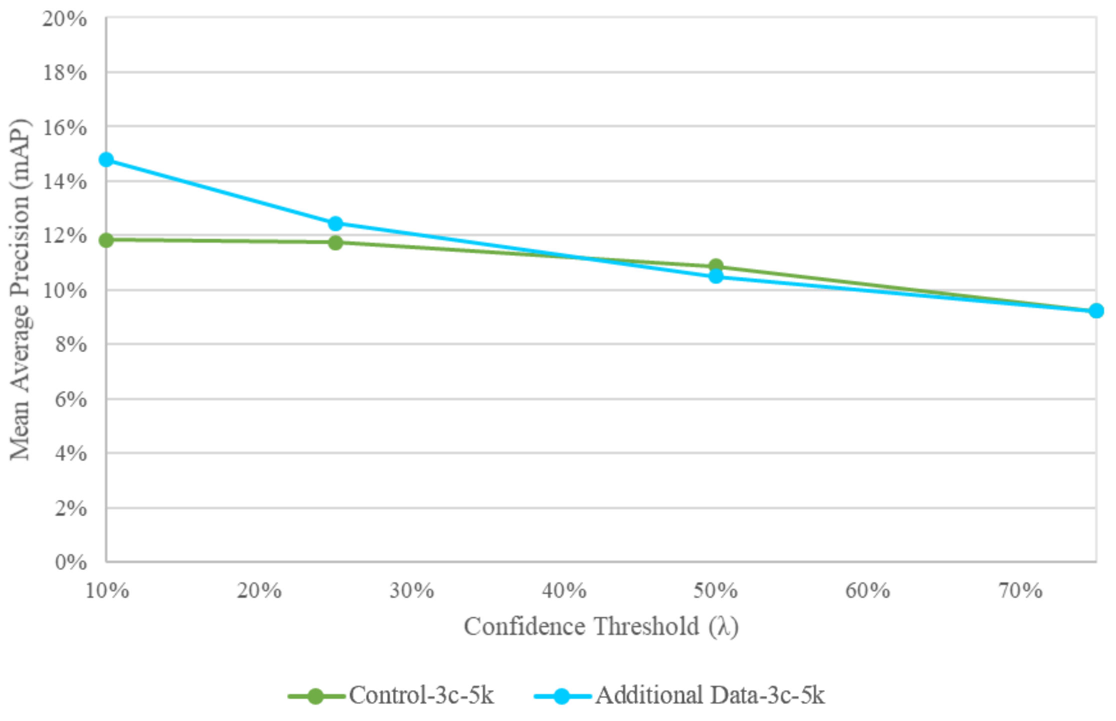


Figure 4.2.3 - IOU 75% Overfitting Evaluation

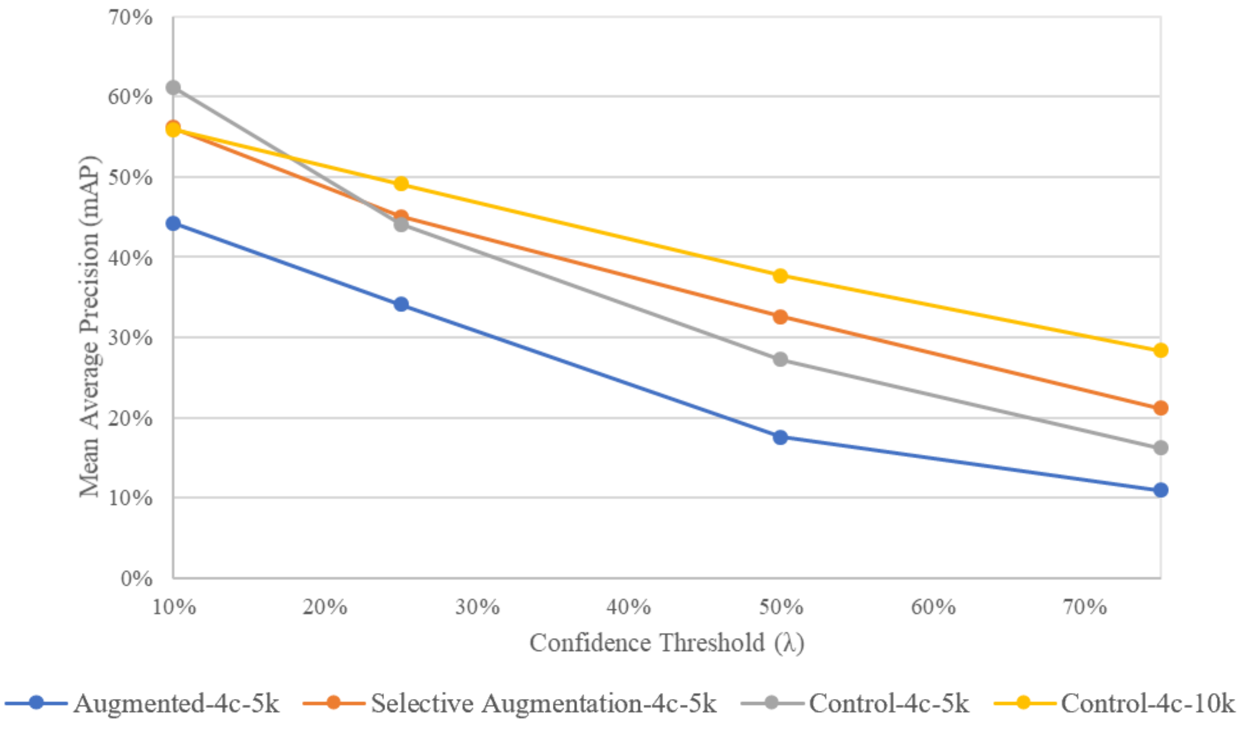


Figure 4.2.4 - IOU 35% Augmentation and Step Evaluation

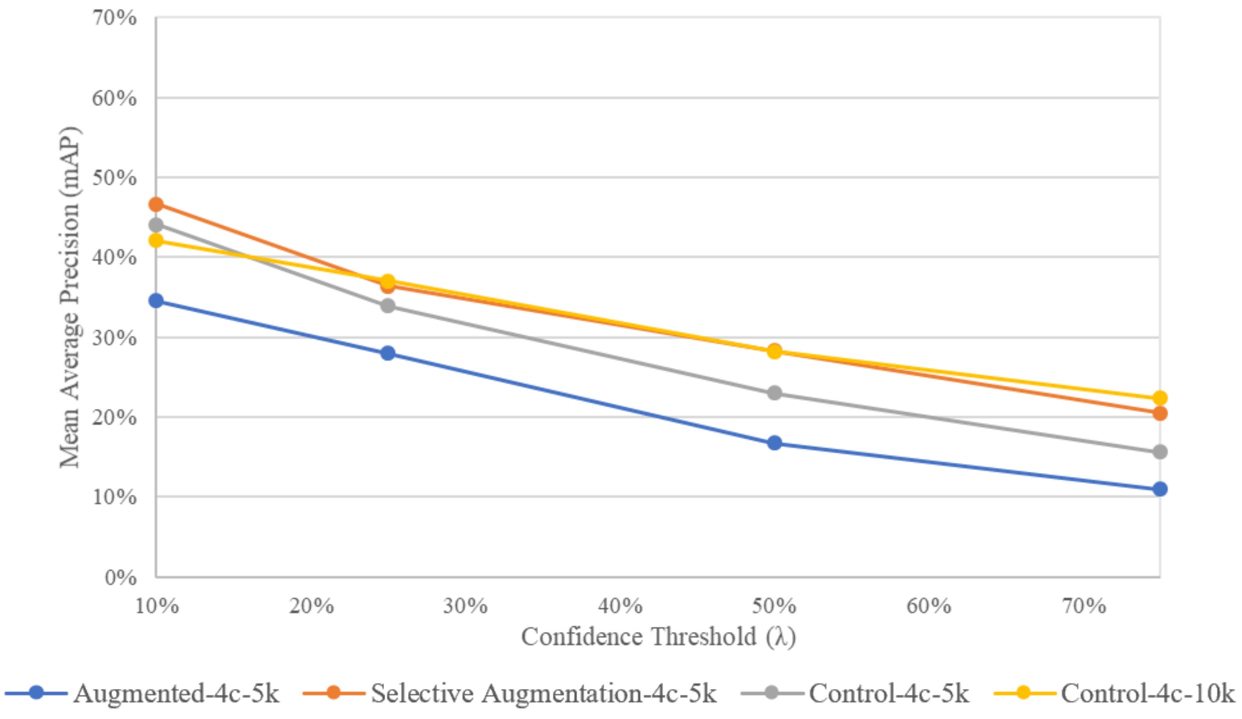


Figure 4.2.5 - IOU 50% Augmentation and Step Evaluation

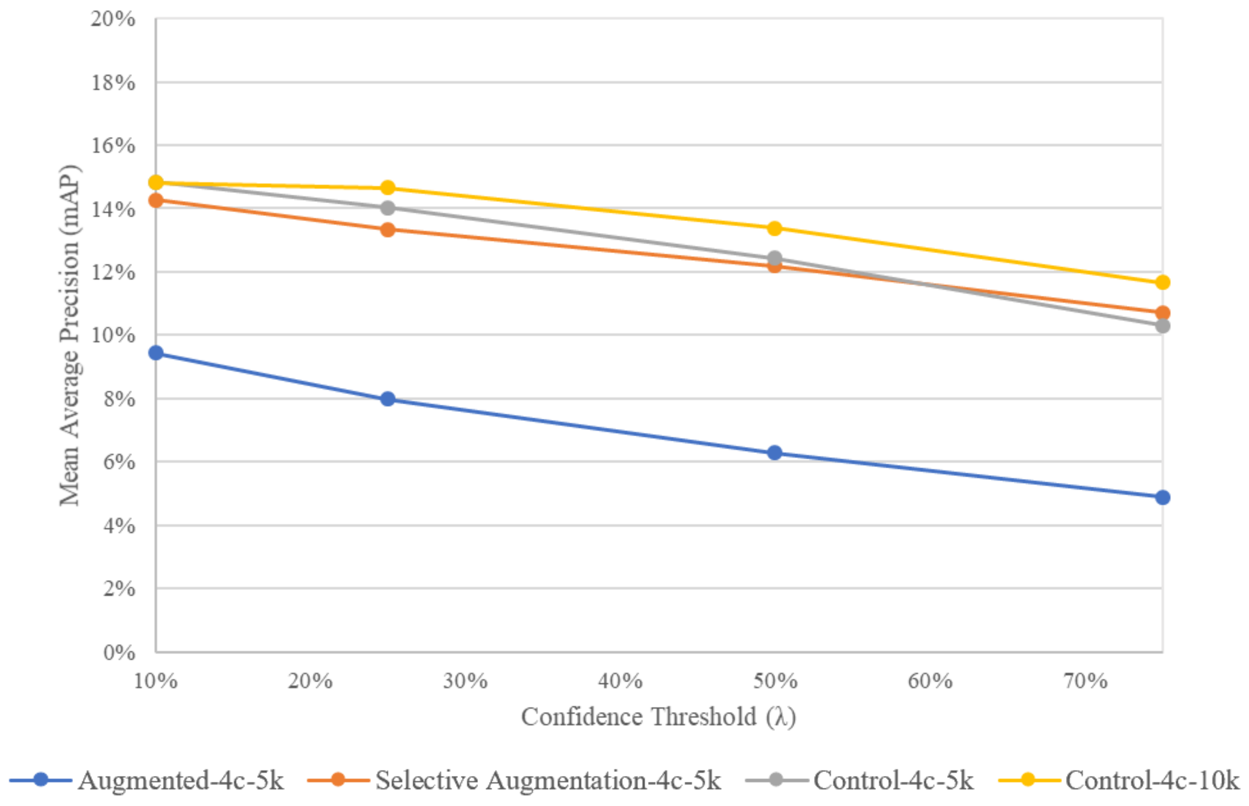


Figure 4.2.6 - IOU 75% Augmentation and Step Evaluation

Table 4.2.1 – mAP Difference of Selective Augmentation-4c-5k and Control-4c-5k

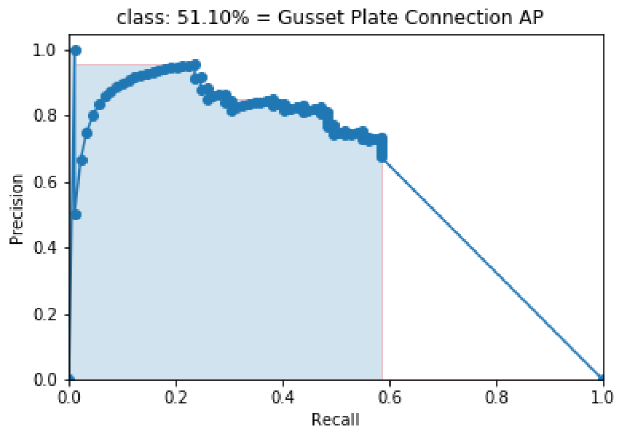
	λ^{10}	λ^{25}	λ^{50}	λ^{75}
AP^{35}	-5.1%	0.9%	5.4%	5.0%
AP^{50}	2.6%	2.5%	5.3%	4.9%
AP^{75}	-0.6%	-0.7%	-0.2%	0.4%
Where the negative values or grey-filled cells indicate where the Control-4c-5k had greater mAP, and the positive or orange-filled cells indicate where the Selective Augmentation-4c-5k had the greater mAP.				

Table 4.2.2 – mAP Difference of Control-4c-5k and Control-4c-10k

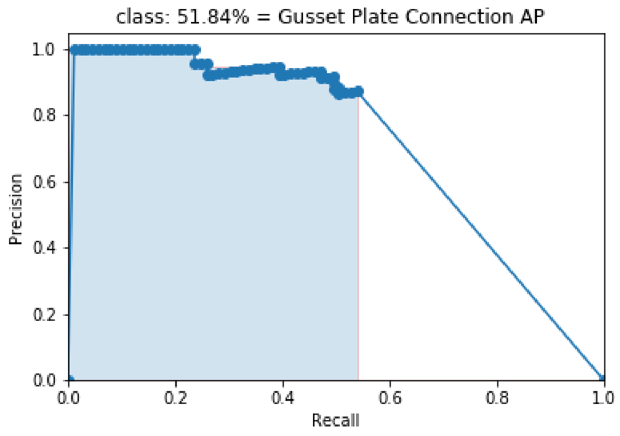
	λ^{10}	λ^{25}	λ^{50}	λ^{75}
AP^{35}	5.2%	-5.1%	-10.5%	-12.2%
AP^{50}	2.0%	-3.1%	-5.3%	-6.7%
AP^{75}	0.0%	-0.6%	-1.0%	-1.4%

Where the positive values or grey-filled cells indicate where the Control-4c-5k had greater mAP, and the negative or gold-filled cells indicate where the Control-4c-10k had the greater mAP.

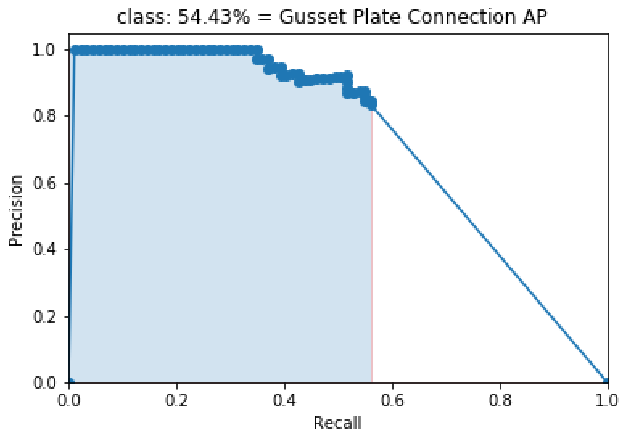
4.3 Recall Curves



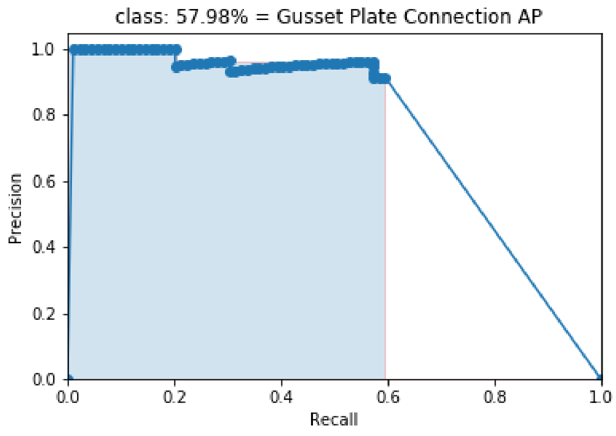
A



B

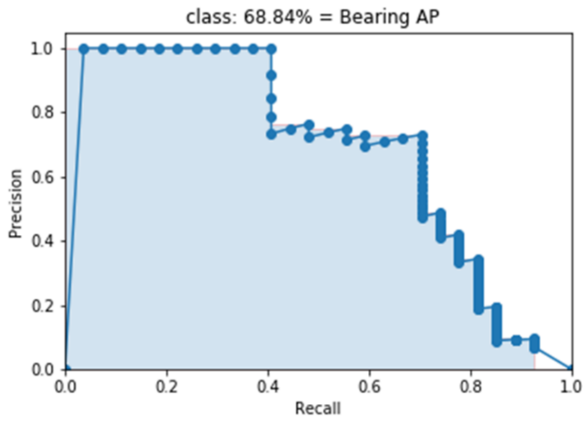


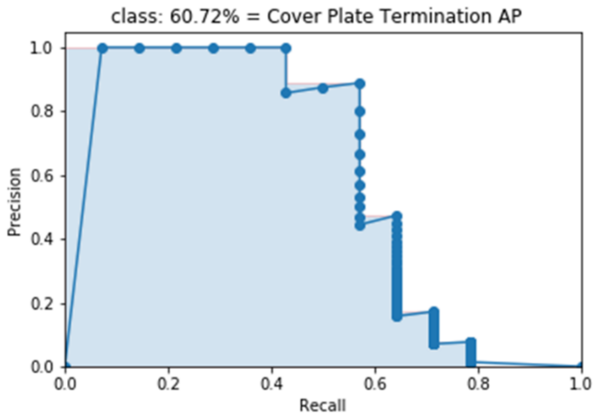
C



D

Figure 4.3.1 – IOU 50%, λ of 25%, Class Recall Curve for Augmented-4c-5k (A), Selective Augmentation-4c-5k (B), Control-4c-5k (C), Control-4c-10k (D)





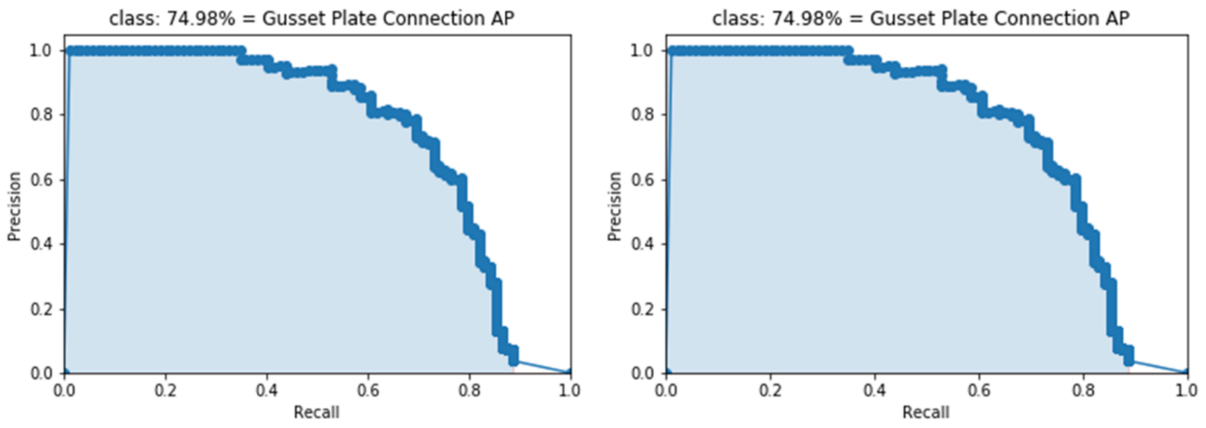


Figure 4.3.2 – IOU 35%, λ of 1%, Class Recall Curve (Control-4c-5k)

4.4 Precision by Structural Detail

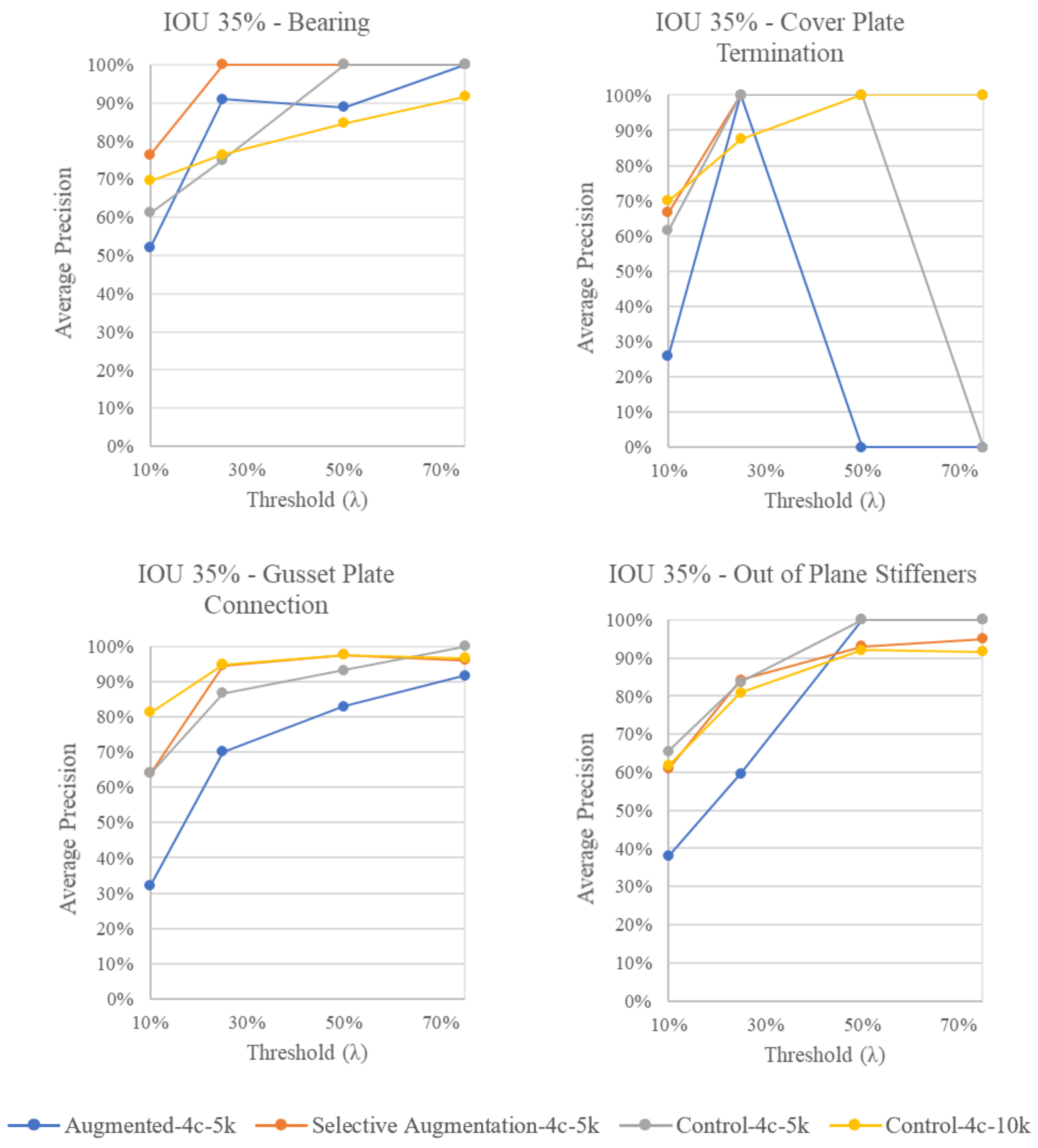


Figure 4.4.1 – IOU 35% Precision by Class

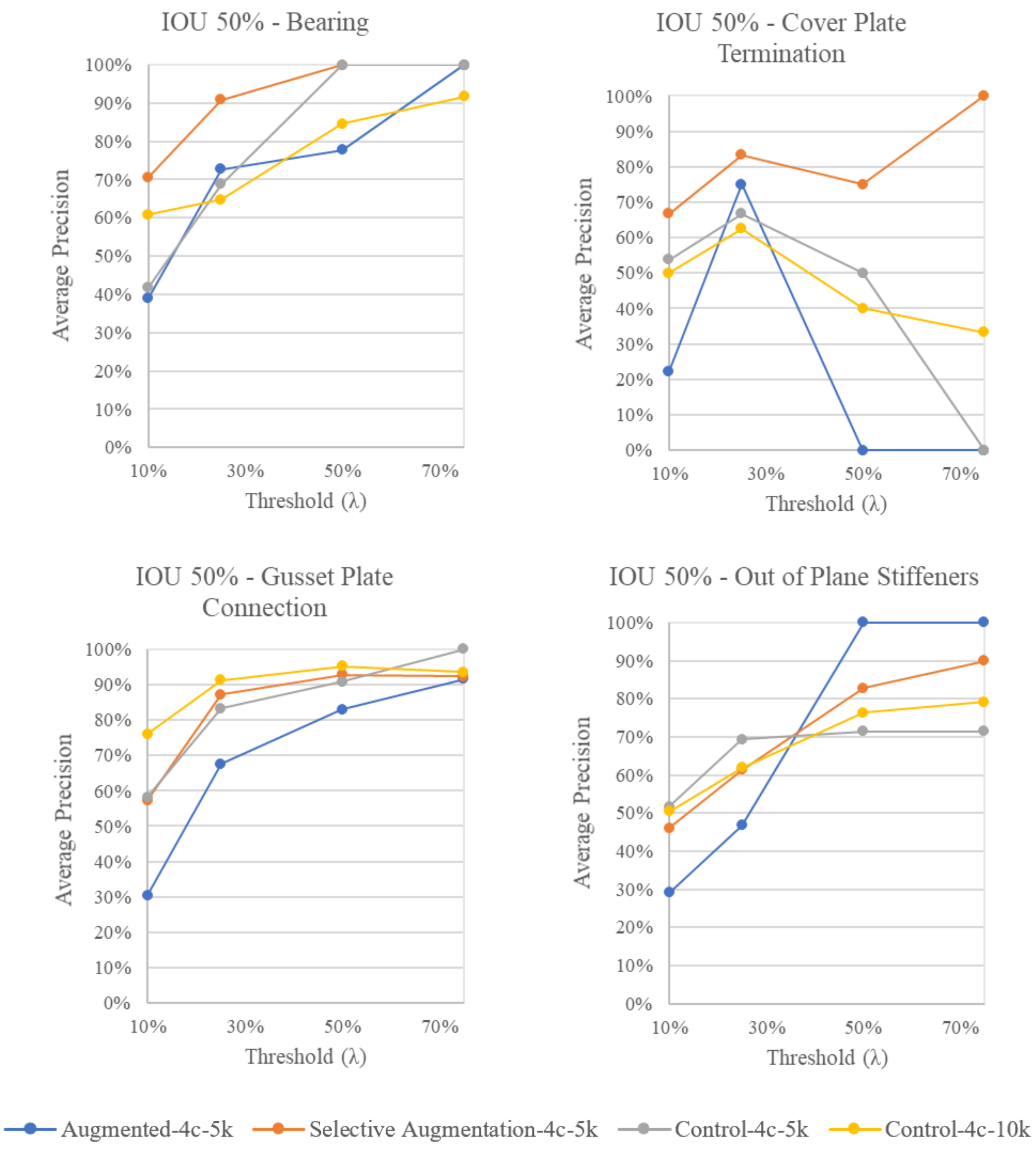

IOU 50% - Bearing
IOU 50% - Cover Plate Termination
IOU 50% - Gusset Plate Connection
IOU 50% - Out of Plane Stiffeners
Average Precision
Threshold (λ)
0%
10%
20%
30%
40%
50%
60%
70%
80%
90%
100%
Augmented-4c-5k
Selective Augmentation-4c-5k
Control-4c-5k
Control-4c-10k

Figure 4.4.2 – IOU 50% Precision by Class

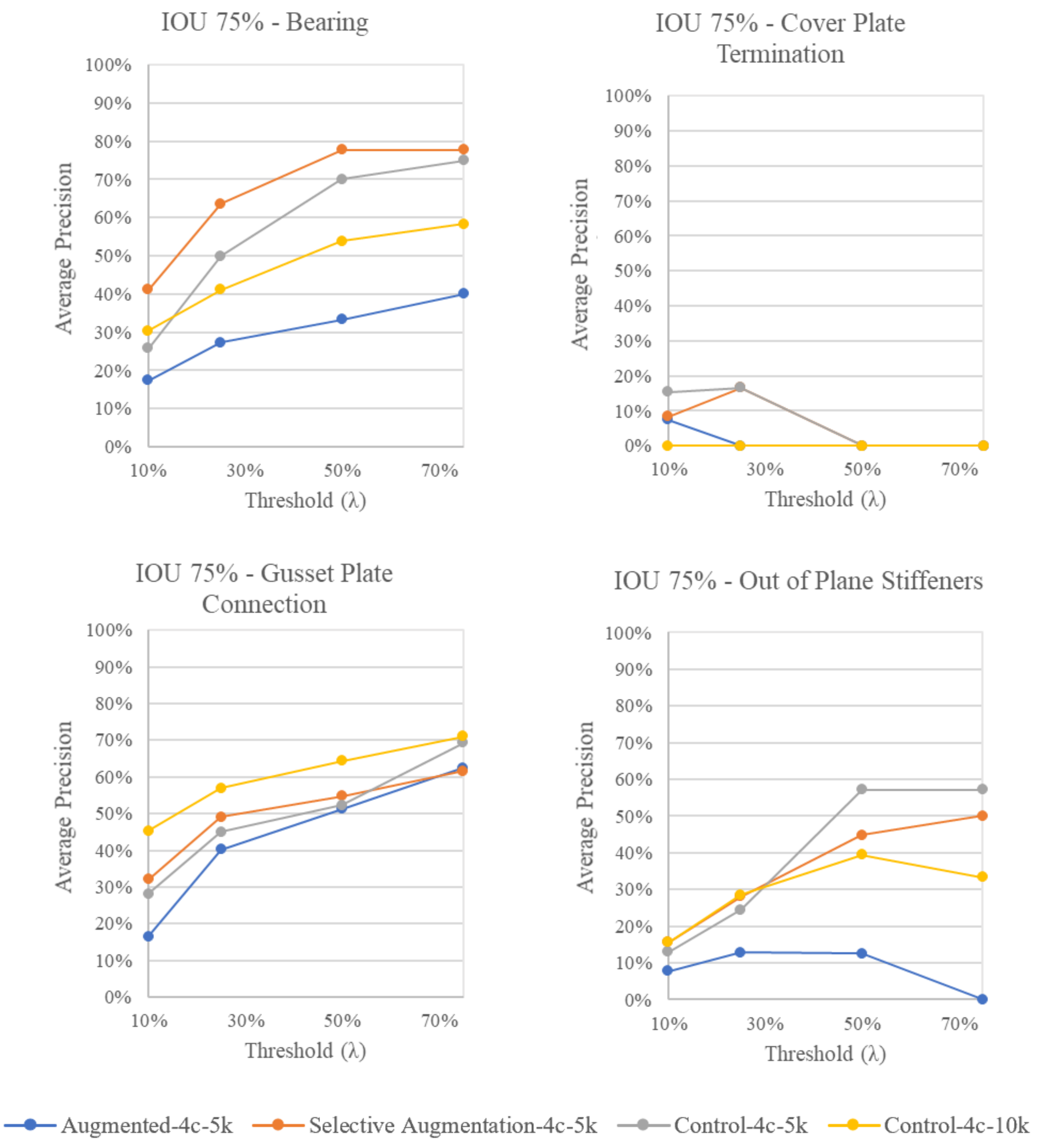


Figure 4.4.3 – IOU 75% Precision by Class

Table 4.4.1 - Average Percentage Loss of Precision in All Models Between IOU Thresholds

IOU	**Bearing**	**Cover Plate Termination**	**Gusset Plate Connection**	**Out of Plane Stiffener**
35%-50%	8.10%	22.54%	4.25%	15.73%
50%-75%	30.56%	73.79%	32.68%	46.60%

$$\frac{\sum_{Model\ 1}^{Model\ 4}\sum_{10\%}^{75\%}\frac{(IOU_{35}\lambda_i)-(IOU_{50}\lambda_i)}{(IOU_{35}\lambda_i)}}{16}$$

Where there are four models (1-4), and each model took the percent drop in precision from the subsequent IOU at each confidence threshold (λ). This method captured the overall decline in precision.

4.5 Average Class Precision

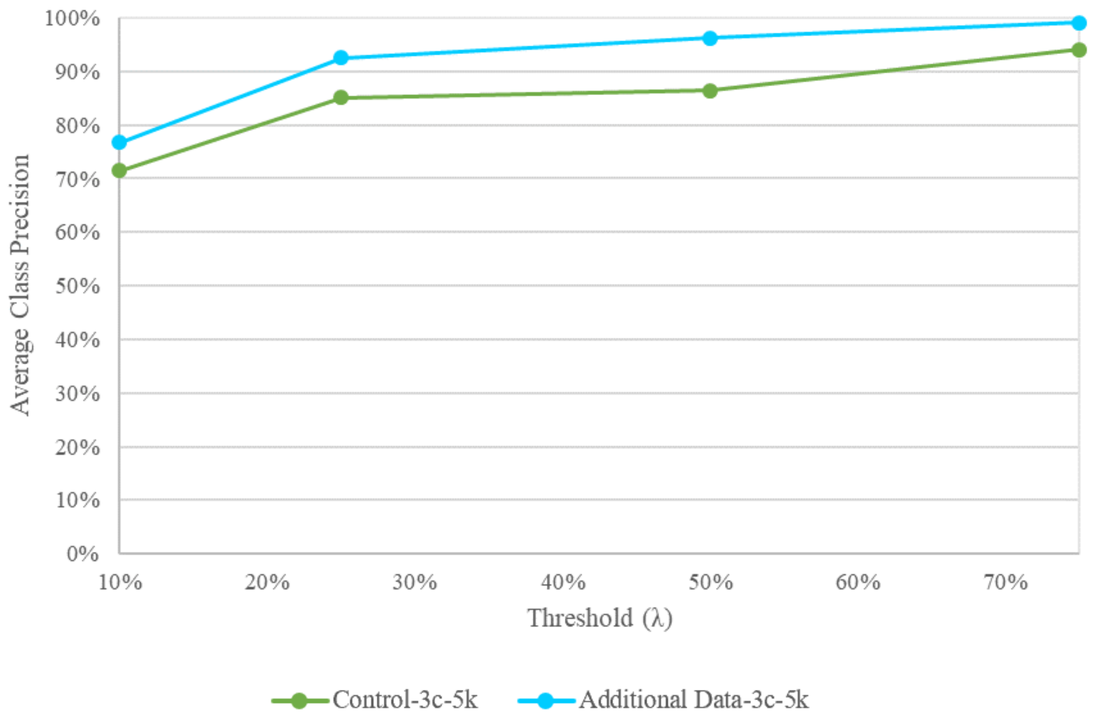


Figure 4.5.1 - IOU 35% Overfitting Evaluation

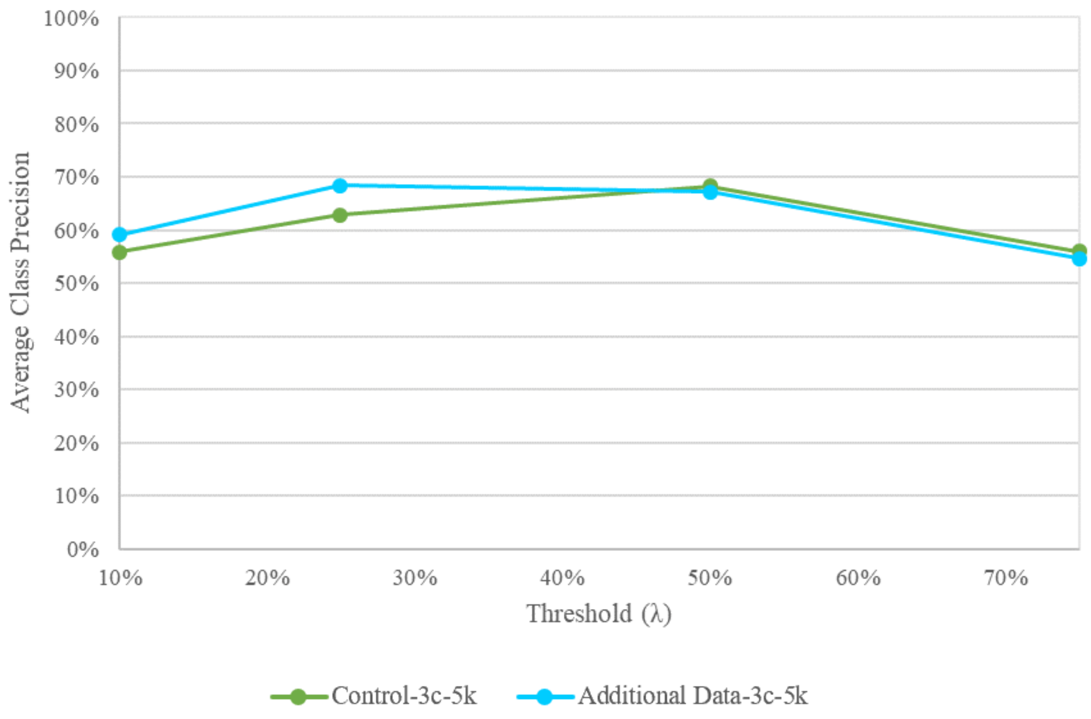


Figure 4.5.2 - IOU 50% Overfitting Evaluation

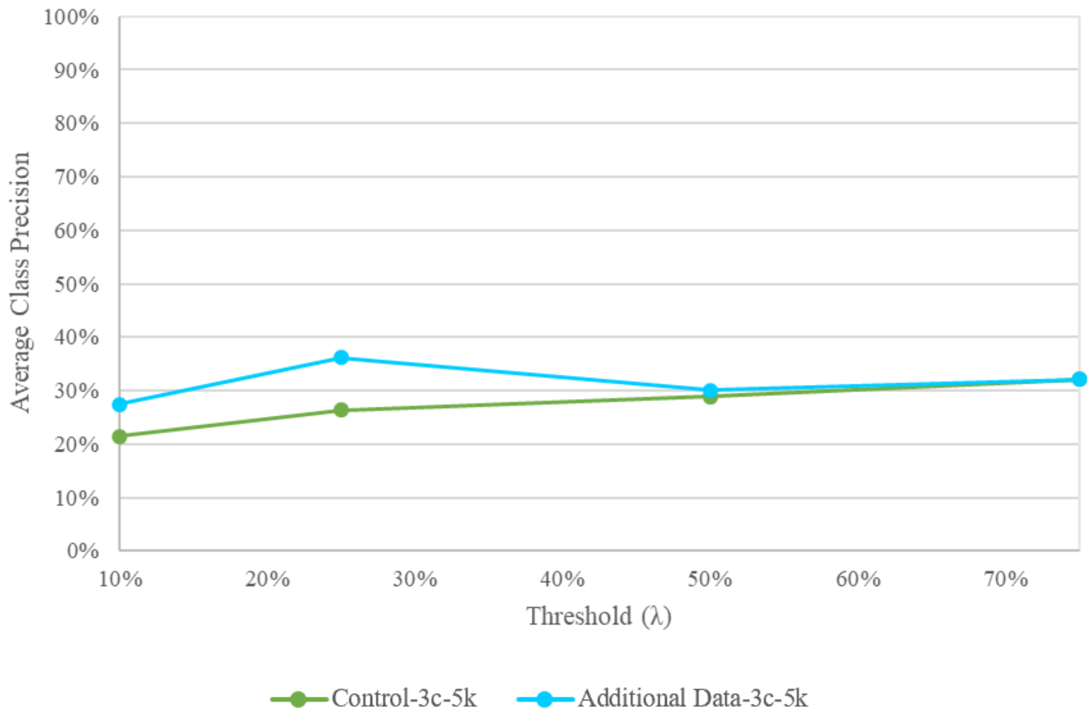


Figure 4.5.3 - IOU 75% Overfitting Evaluation

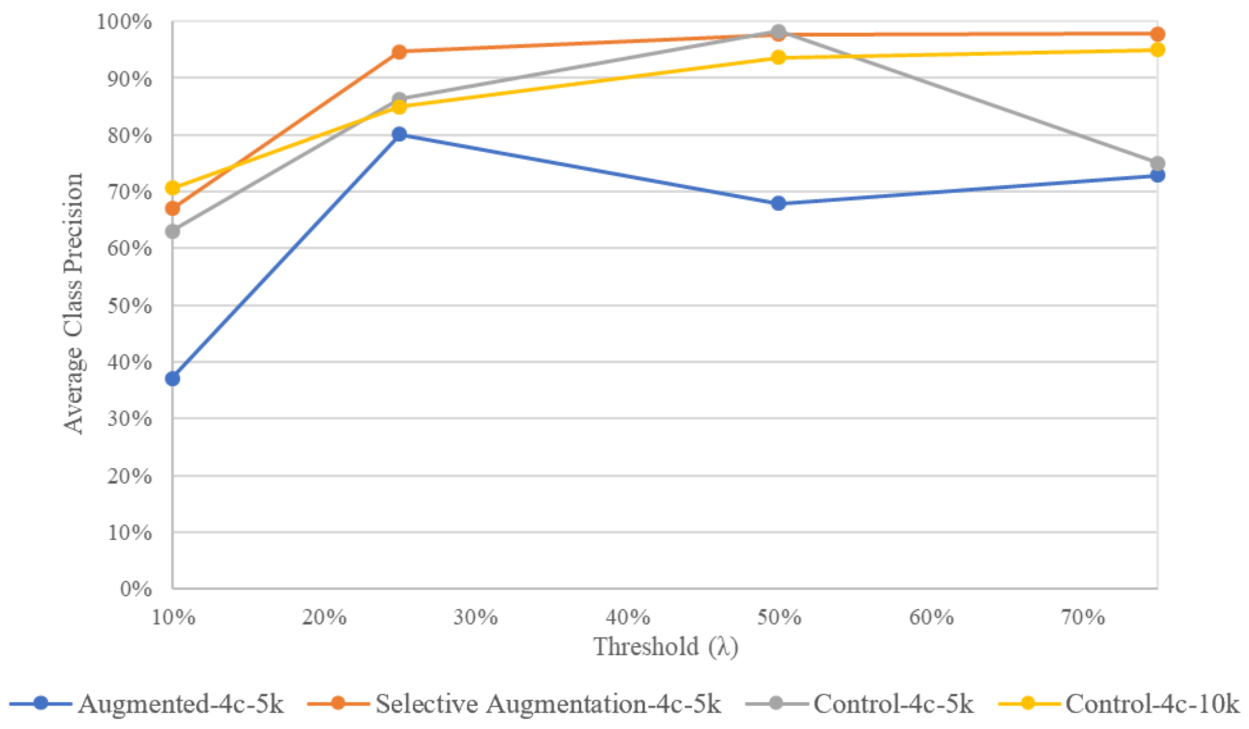


Figure 4.5.4 - IOU 35% Augmentation and Step Evaluation

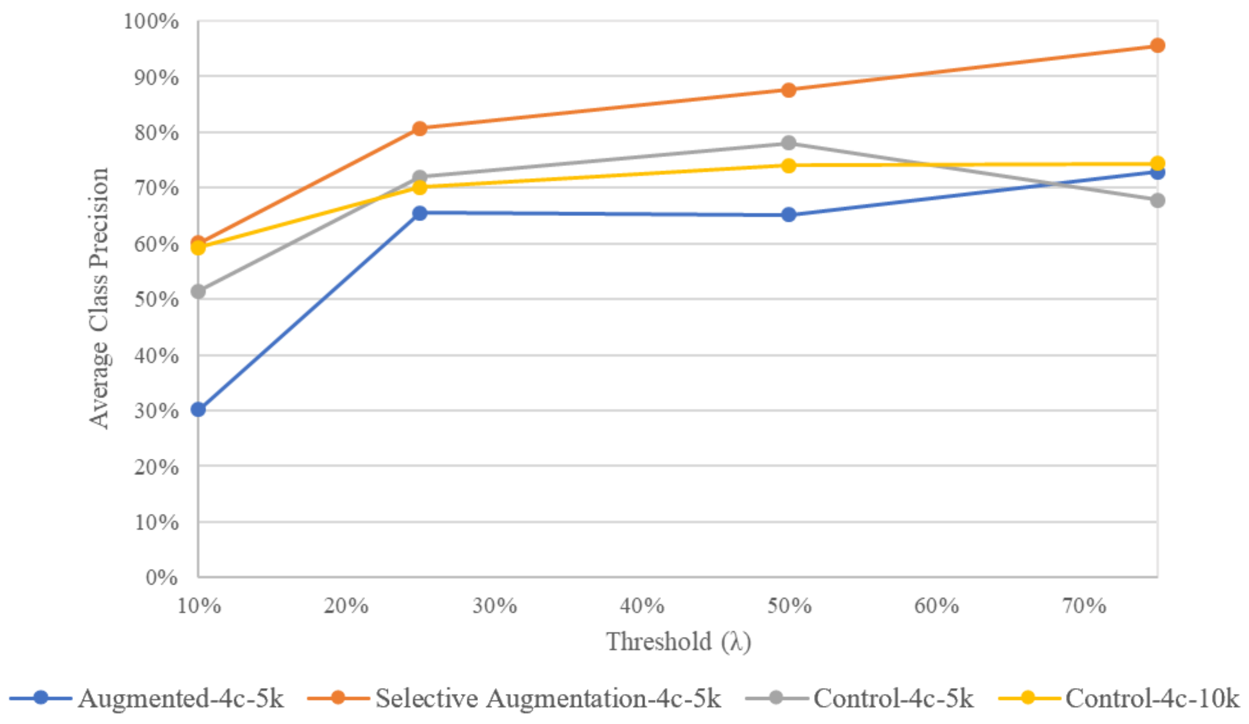


Figure 4.5.5 - IOU 50% Augmentation and Step Evaluation

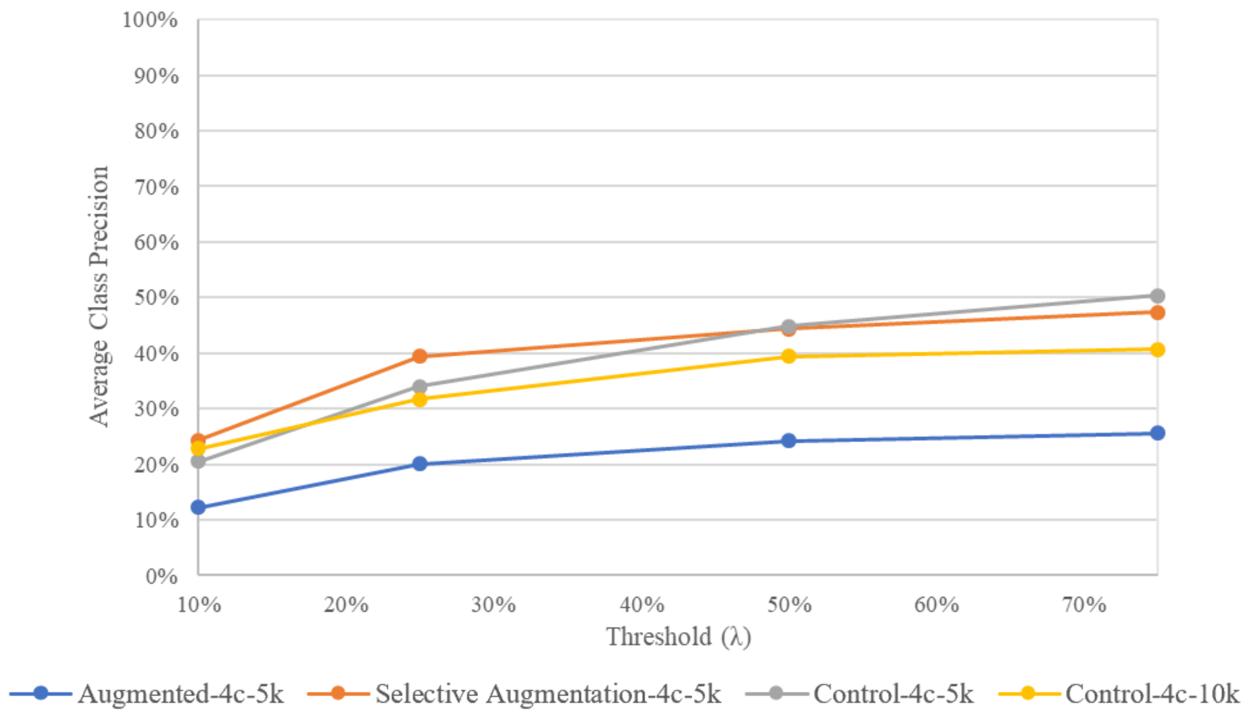


Figure 4.5.6 – IOU 75% Augmentation and Step Evaluation

Table 4.5.1 – Precision Difference of Selective Augmentation-4c-5k and Control-4c-5k

	λ^{10}	λ^{25}	λ^{50}	λ^{75}
AP^{35}	3.9%	8.4%	-0.6%	22.8%
AP^{50}	8.7%	8.7%	9.6%	27.7%
AP^{75}	3.7%	5.3%	-0.5%	-3.0%

Where the negative values or grey-filled cells indicate where the Control-4c-5k had greater precision, and the positive or orange-filled cells indicate where the Selective Augmentation-4c-5k had the greater precision.

Table 4.5.2 – Precision Difference of Control-4c-5k and Control-4c-10k

	λ^{10}	λ^{25}	λ^{50}	λ^{75}
AP^{35}	-7.6%	1.4%	4.7%	-20.0%
AP^{50}	-7.9%	1.9%	4.0%	-6.6%
AP^{75}	-2.3%	2.4%	5.5%	9.7%
Where the positive values or grey-filled cells indicate where the Control-4c-5k had greater precision, and the negative or gold-filled cells indicate where the Control-4c-10k had the greater precision.				

4.6 Accuracy by Structural Detail

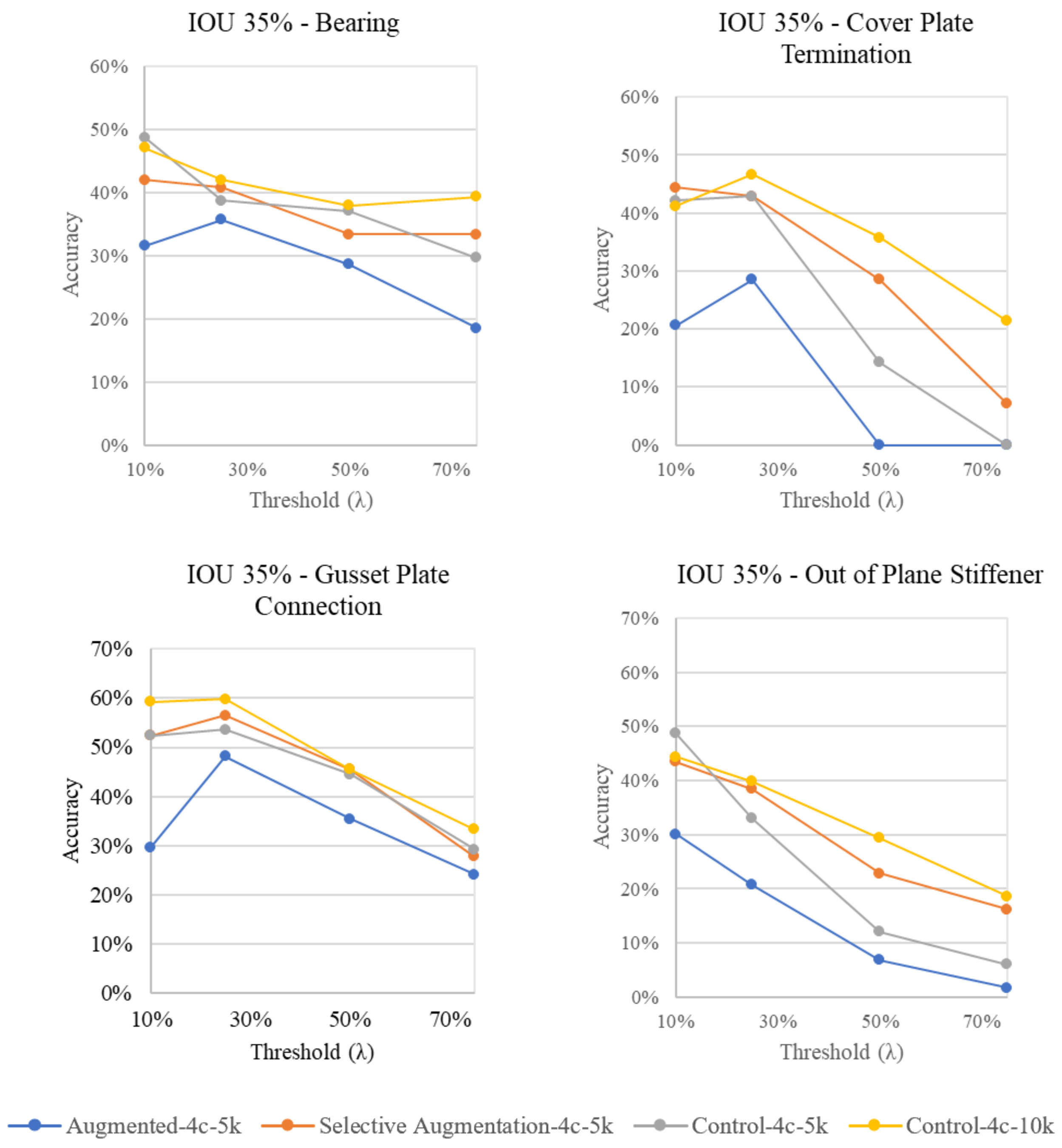


Figure 4.6.1 – IOU 35% Accuracy by Structural Detail

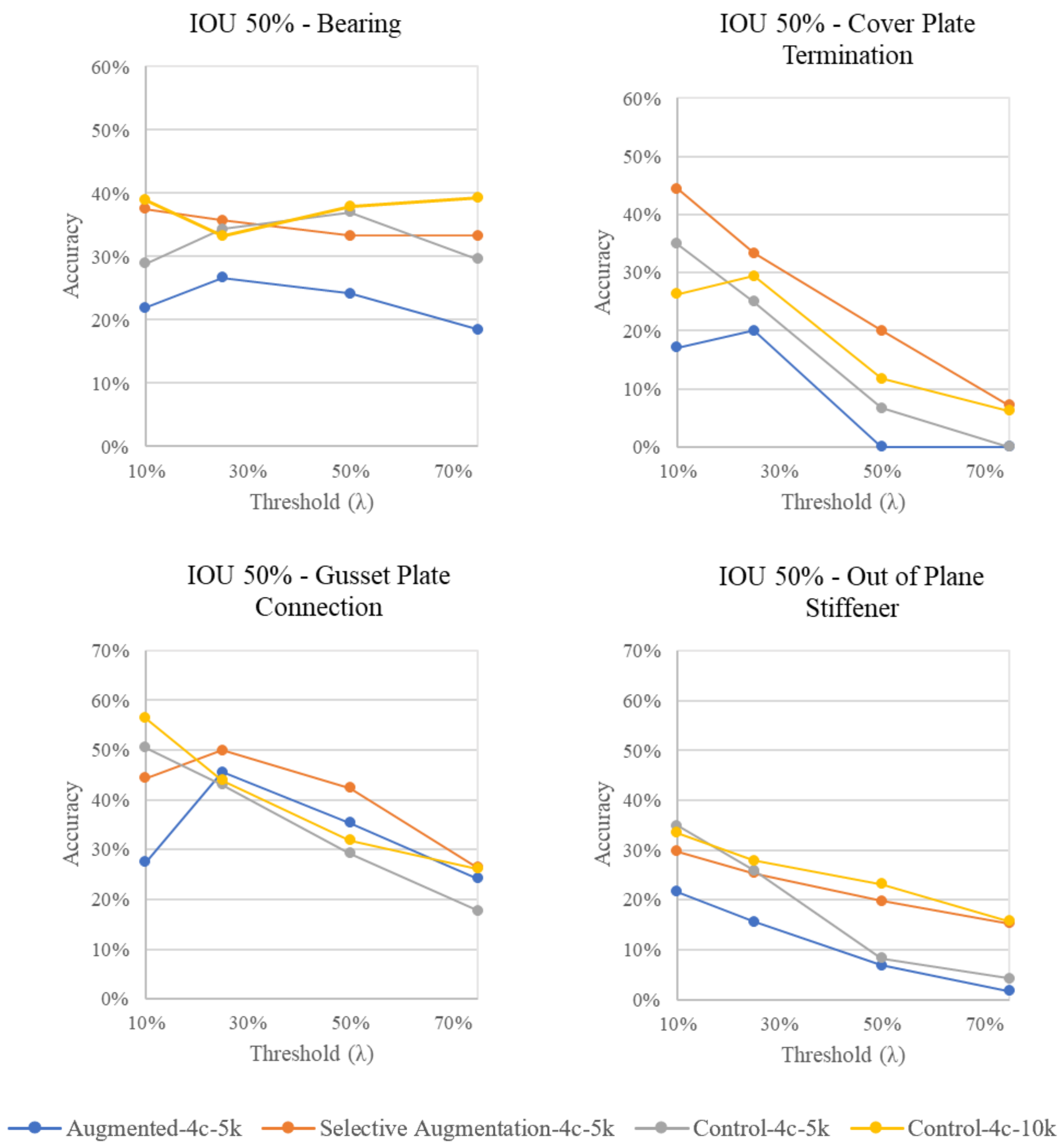

IOU 50% - Bearing
IOU 50% - Cover Plate Termination
IOU 50% - Gusset Plate Connection
IOU 50% - Out of Plane Stiffener
Accuracy
Threshold (λ)
0%
10%
20%
30%
40%
50%
60%
70%
Augmented-4c-5k
Selective Augmentation-4c-5k
Control-4c-5k
Control-4c-10k

Figure 4.6.2 – IOU 50% Accuracy by Structural Detail

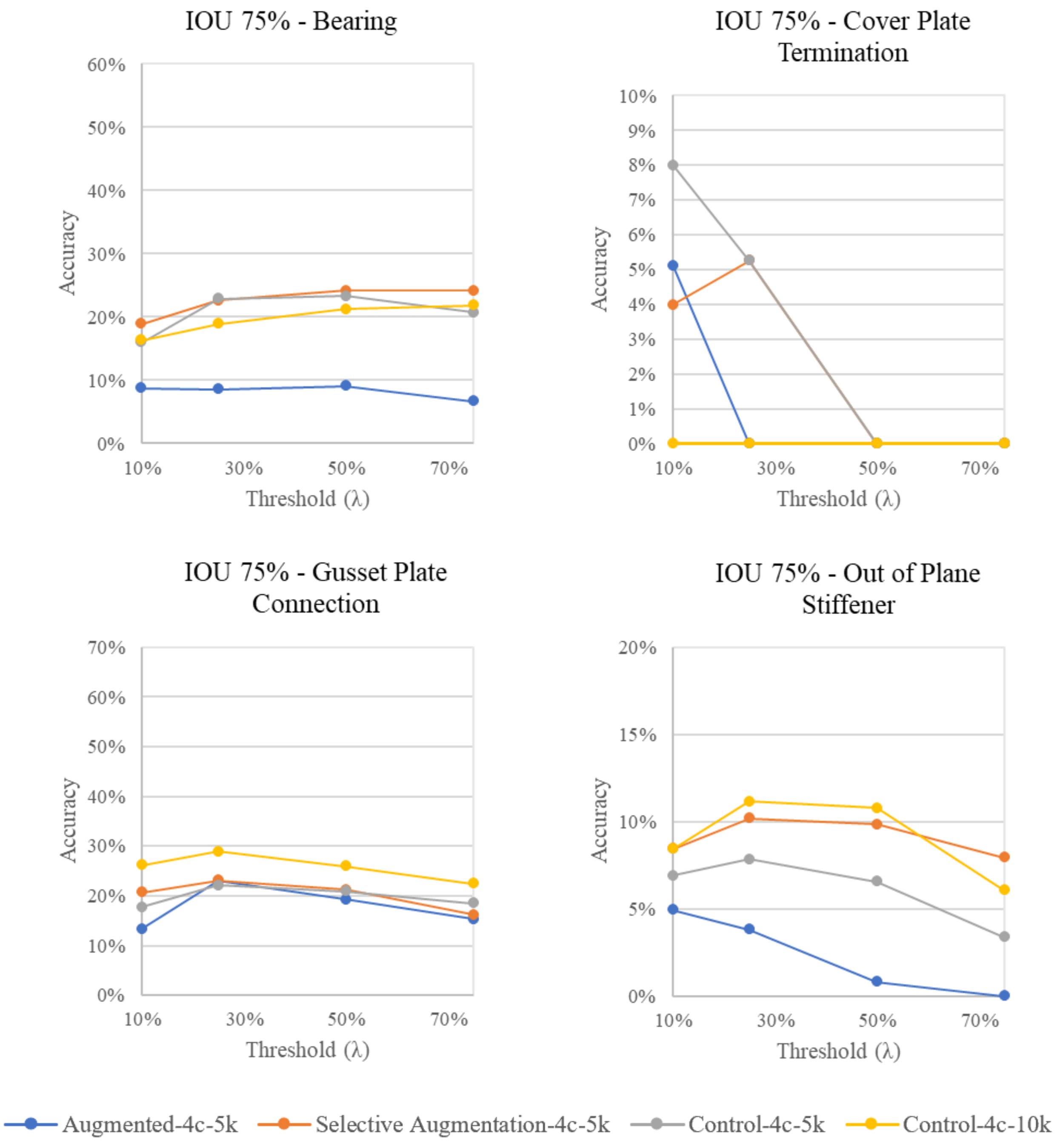


Figure 4.6.3 – IOU 75% Accuracy by Structural Detail

Table 4.6.1 - Average Percentage Loss of Accuracy in All Models Between IOU Thresholds

IOU	**Bearing**	**Cover Plate Termination**	**Gusset Plate Connection**	**Out of Plane Stiffener**
35%-50%	11.5%	26.4%	6.1%	21.2%
50%-75%	24.8%	17.4%	11.5%	28.1%

$$\frac{\sum_{Model\ 1}^{Model\ 4}\sum_{10\%}^{75\%}\frac{(IOU_{35}\lambda_i)-(IOU_{50}\lambda_i)}{(IOU_{35}\lambda_i)}}{16}$$

Where there are four models (1-4), and each model took the percent drop in accuracy from the subsequent IOU at each confidence threshold (λ). This method captured the overall decline in accuracy.

4.7 Average Accuracy

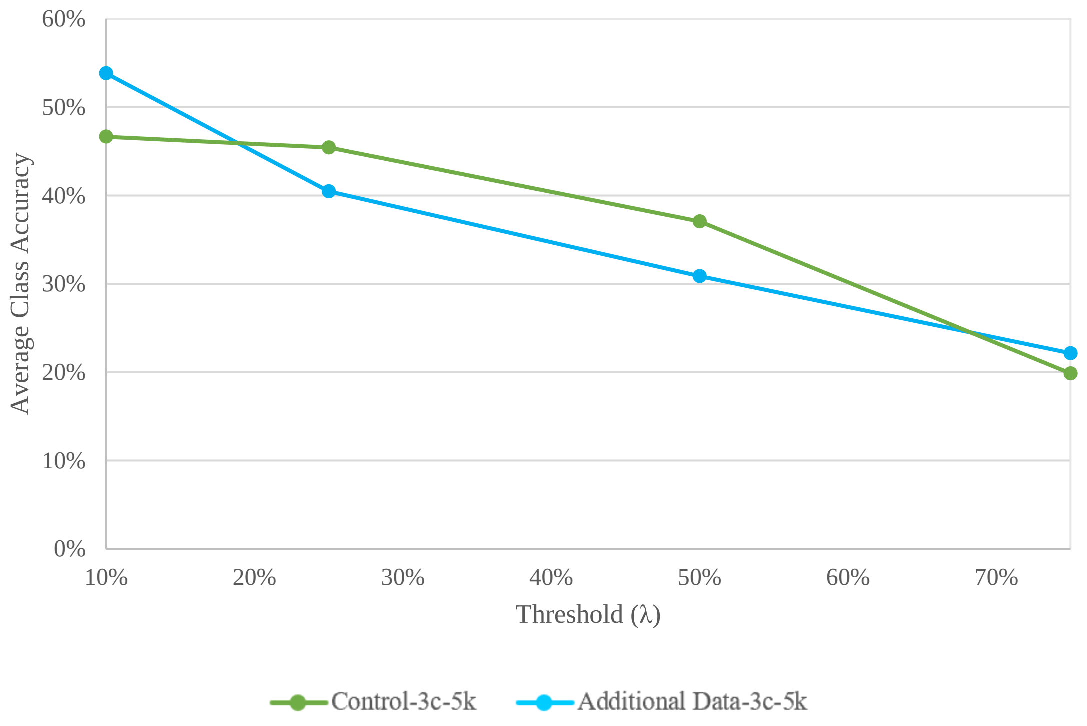


Figure 4.7.1 - IOU 35% Overfitting Evaluation

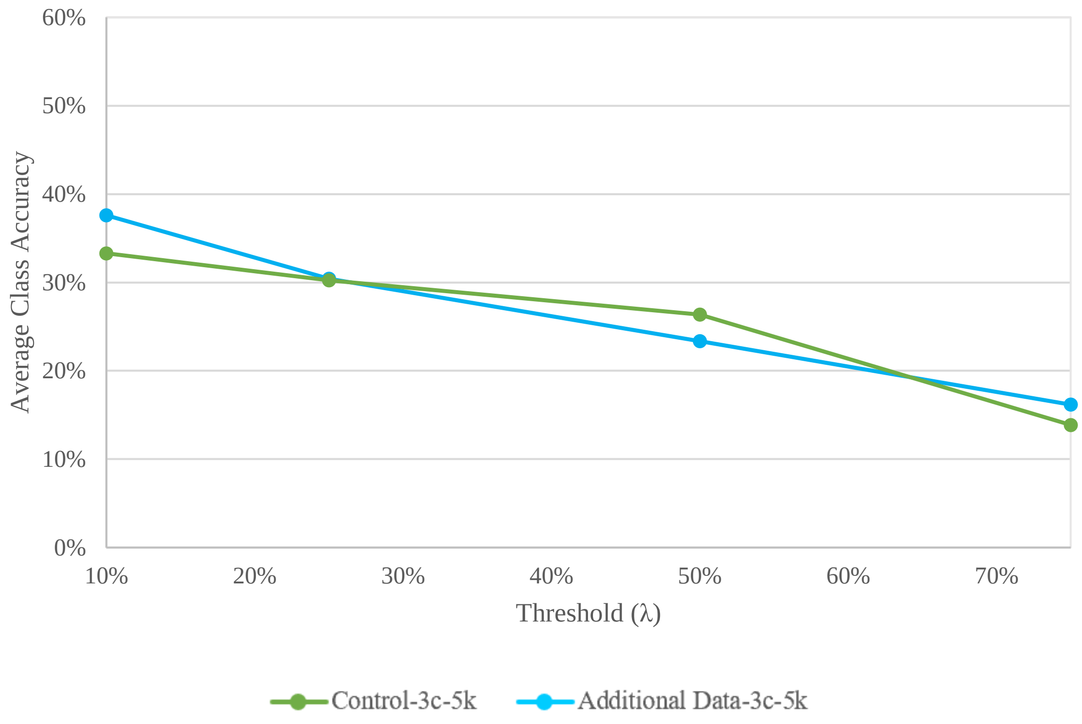


Figure 4.7.2 - IOU 50% Overfitting Evaluation

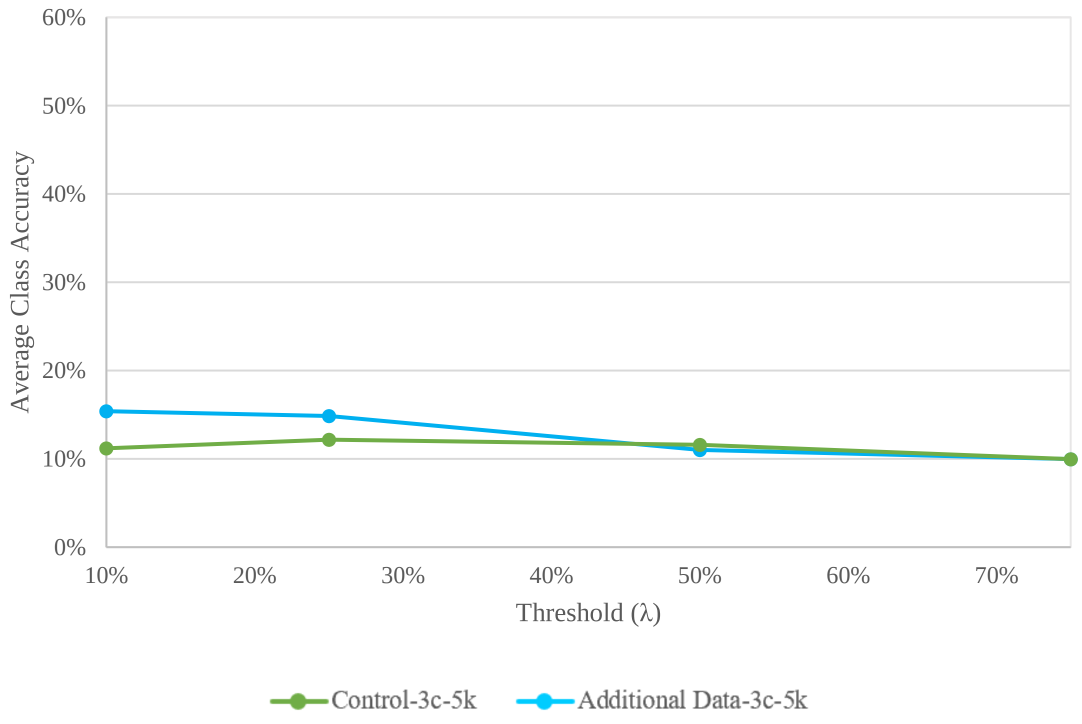


Figure 4.7.3 - IOU 75% Overfitting Evaluation

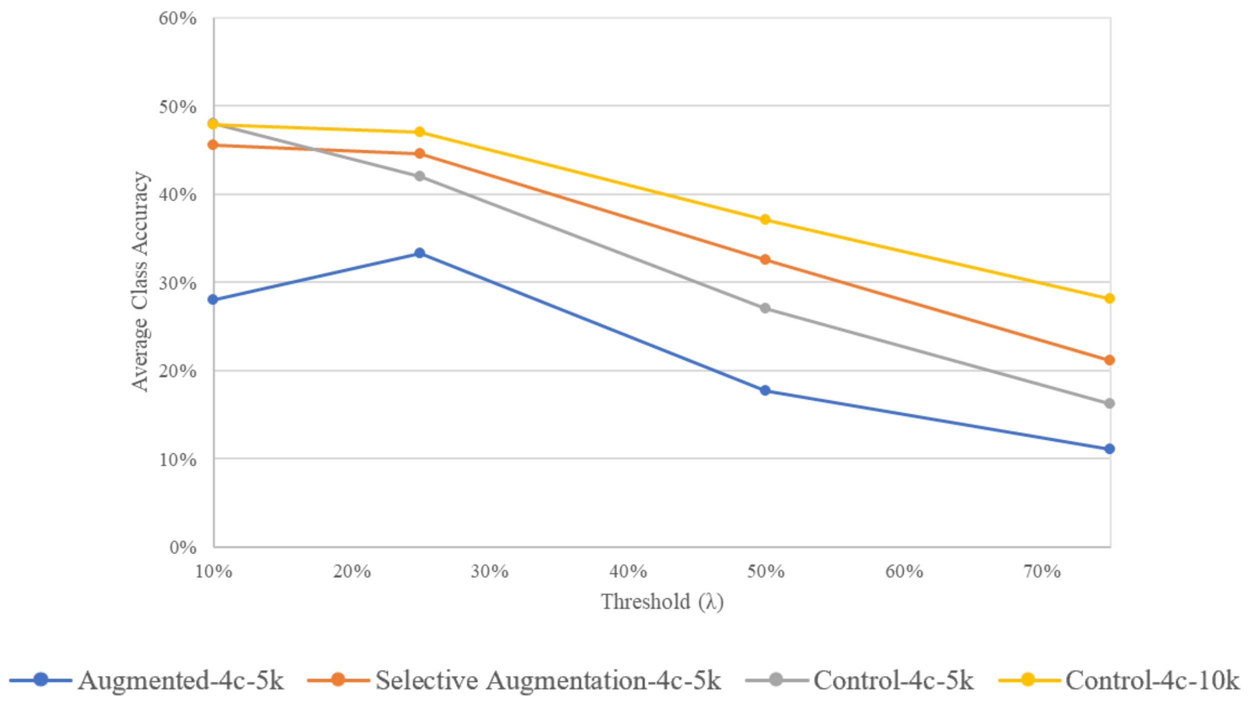


Figure 4.7.4- IOU 35% Augmentation and Step Evaluation

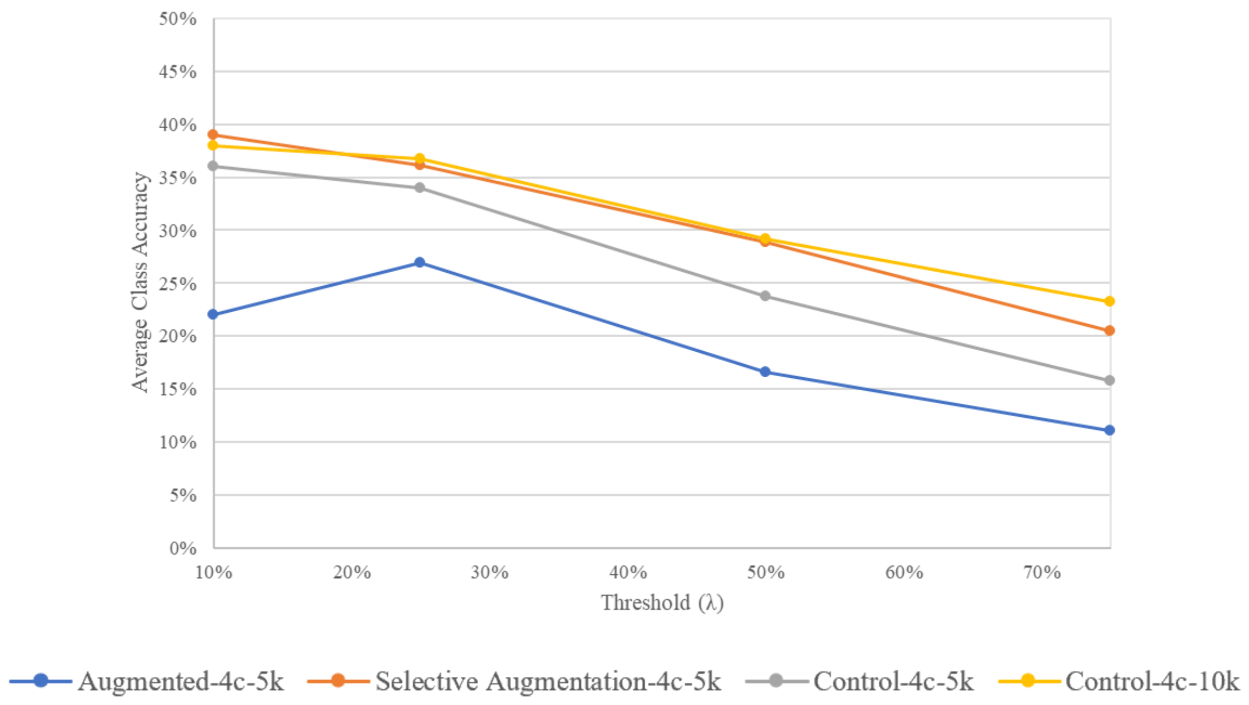


Figure 4.7.5- IOU 50% Augmentation and Step Evaluation

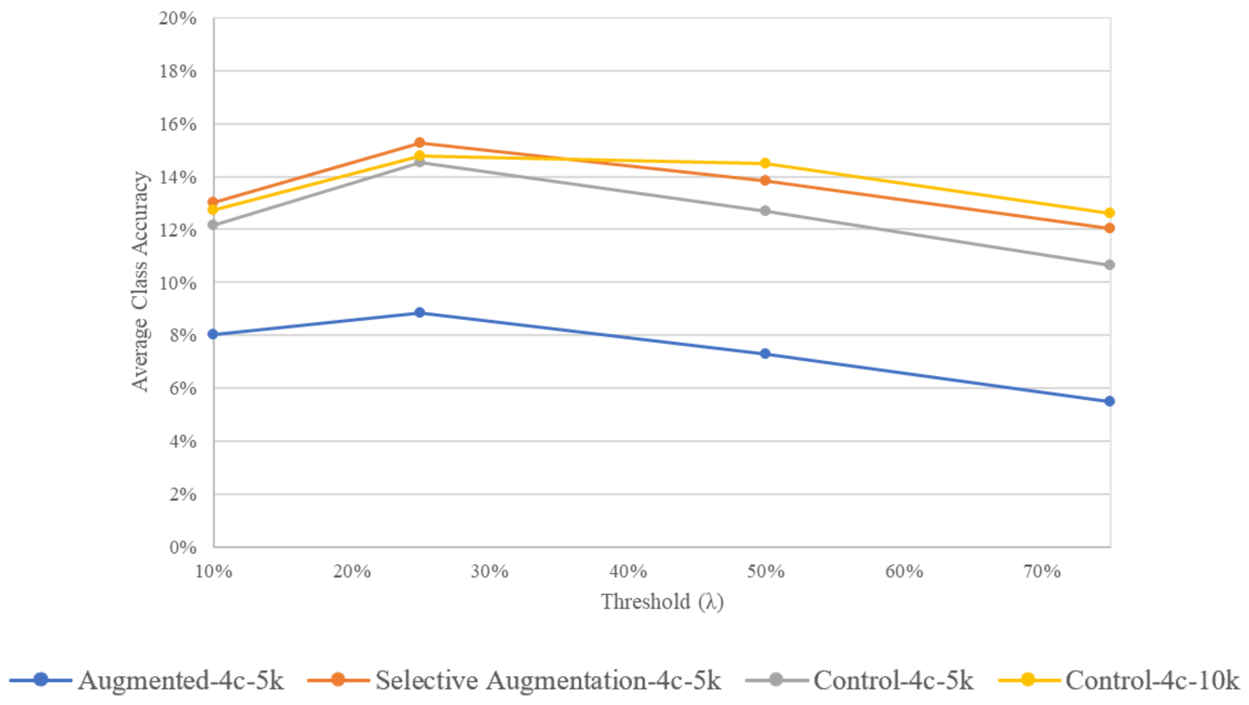


Figure 4.7.6- IOU 75% Augmentation and Step Evaluation

Table 4.7.1 – Accuracy Difference of Selective Augmentation-4c-5k and Control-4c-5k

	λ^{10}	λ^{25}	λ^{50}	λ^{75}
AP^{35}	-2.4%	2.6%	5.6%	4.9%
AP^{50}	3.0%	2.1%	5.1%	4.8%
AP^{75}	0.9%	0.8%	1.1%	1.4%

Where the negative values or grey-filled cells indicate where the Control-4c-5k had greater accuracy, and the positive or orange-filled cells indicate where the Selective Augmentation-4c-5k had the greater accuracy.

Table 4.7.2 – Accuracy Difference of Control-4c-5k and Control-4c-10k

	λ^{10}	**λ^{25}**	**λ^{50}**	**λ^{75}**
AP^{35}	0.0%	5.0%	10.2%	12.0%
AP^{50}	1.9%	2.8%	5.5%	7.5%
AP^{75}	0.5%	0.2%	1.8%	1.9%

Where the positive values or grey-filled cells indicate where the Control-4c-5k had greater accuracy, and the negative or gold-filled cells indicate where the Control-4c-10k had the greater accuracy.

4.8 F1-Scores

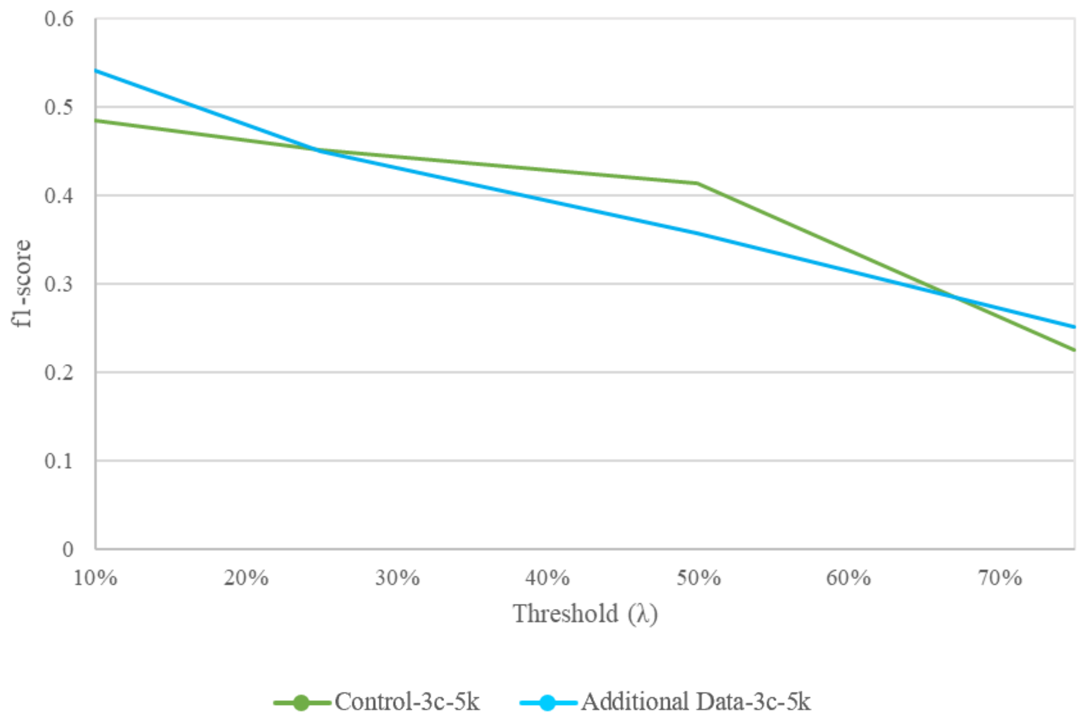


Figure 4.8.1 - IOU 35% Overfitting Evaluation

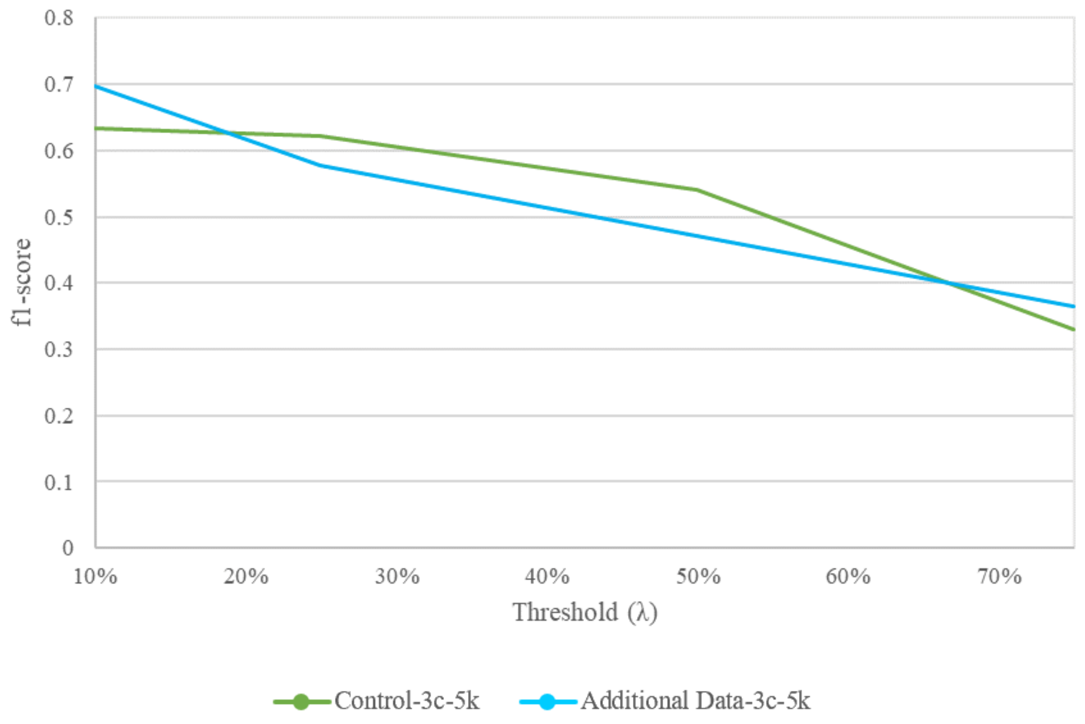


Figure 4.8.2 - IOU 50% Overfitting Evaluation

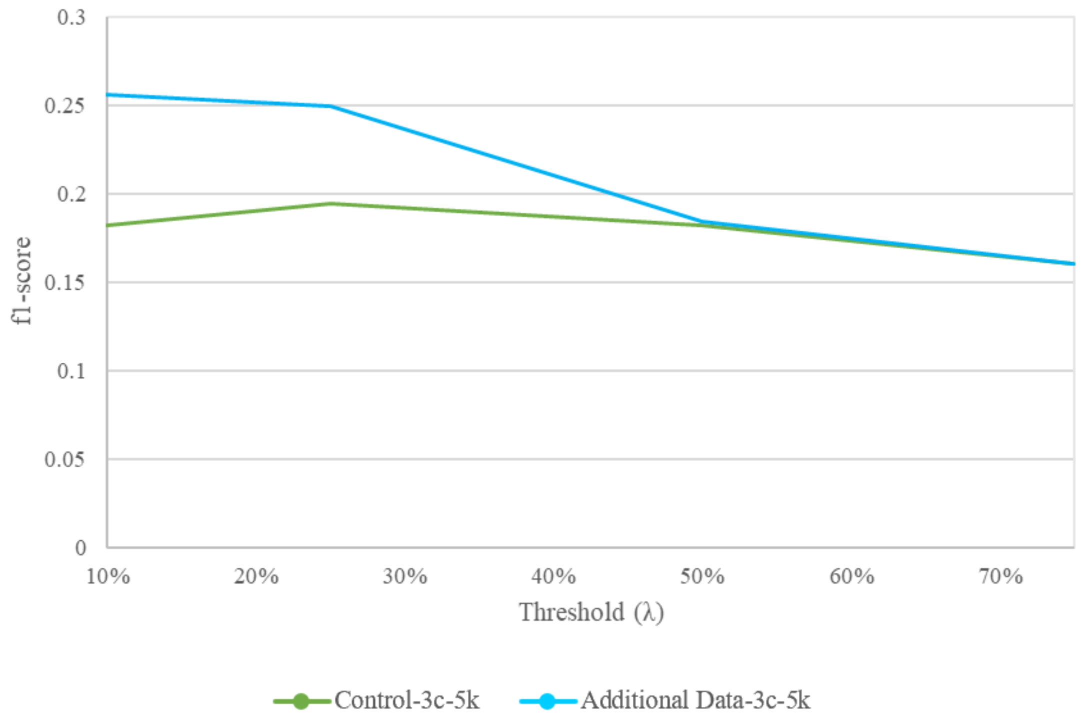


Figure 4.8.3 - IOU 75% Overfitting Evaluation

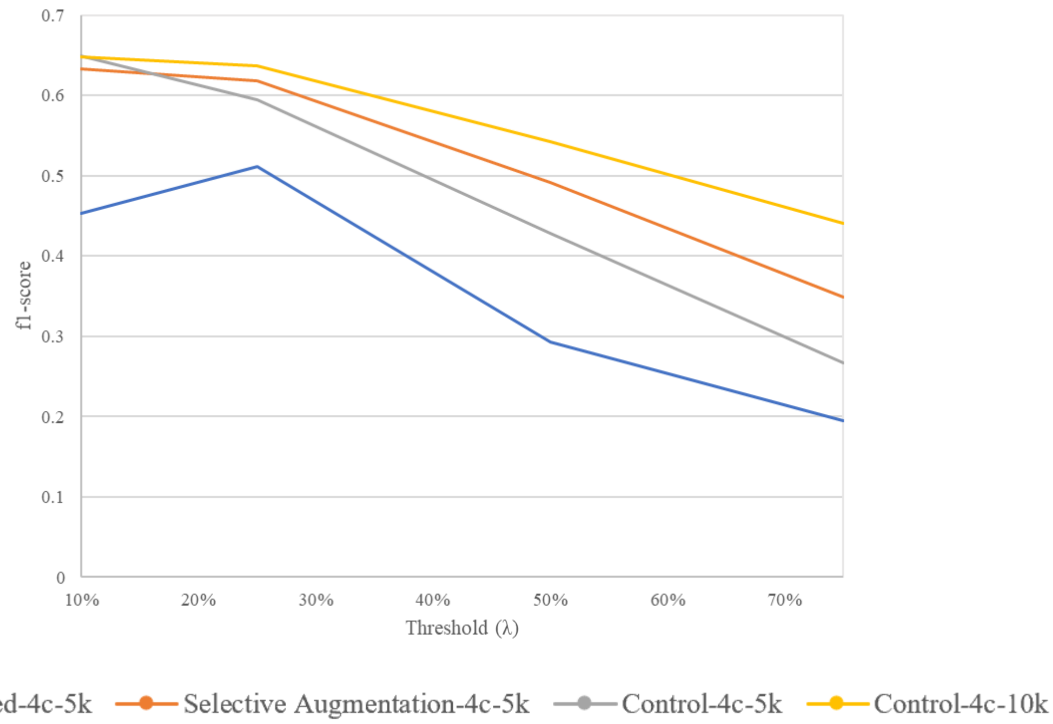


Figure 4.8.4- IOU 35% Augmentation and Step Evaluation

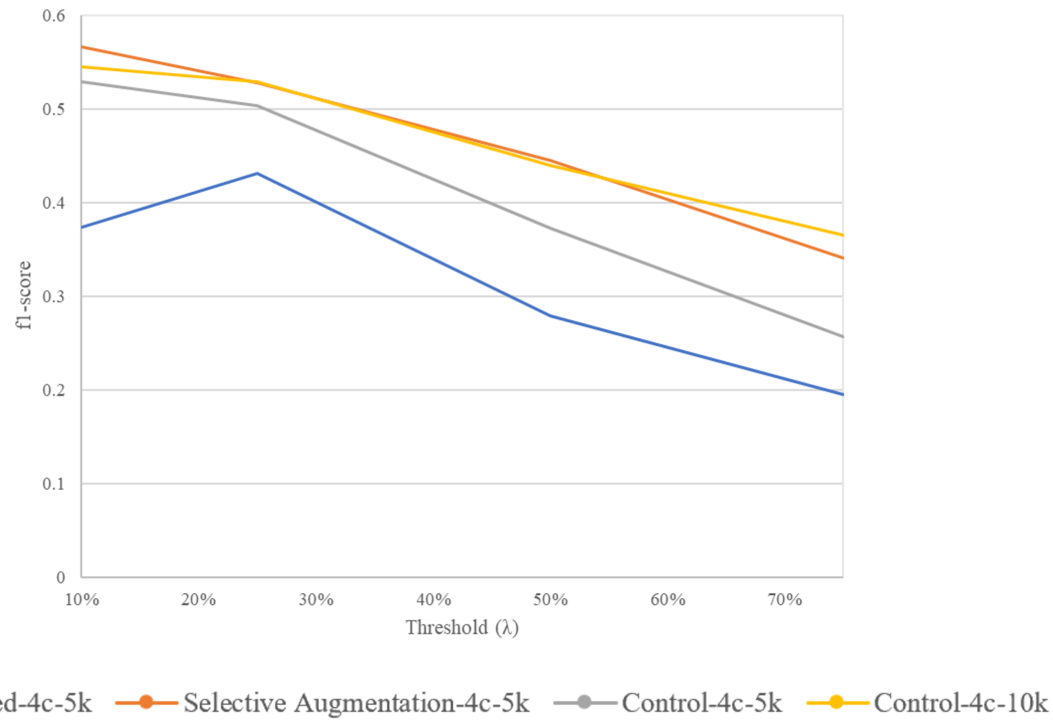


Figure 4.8.5 - IOU 50% Augmentation and Step Evaluation

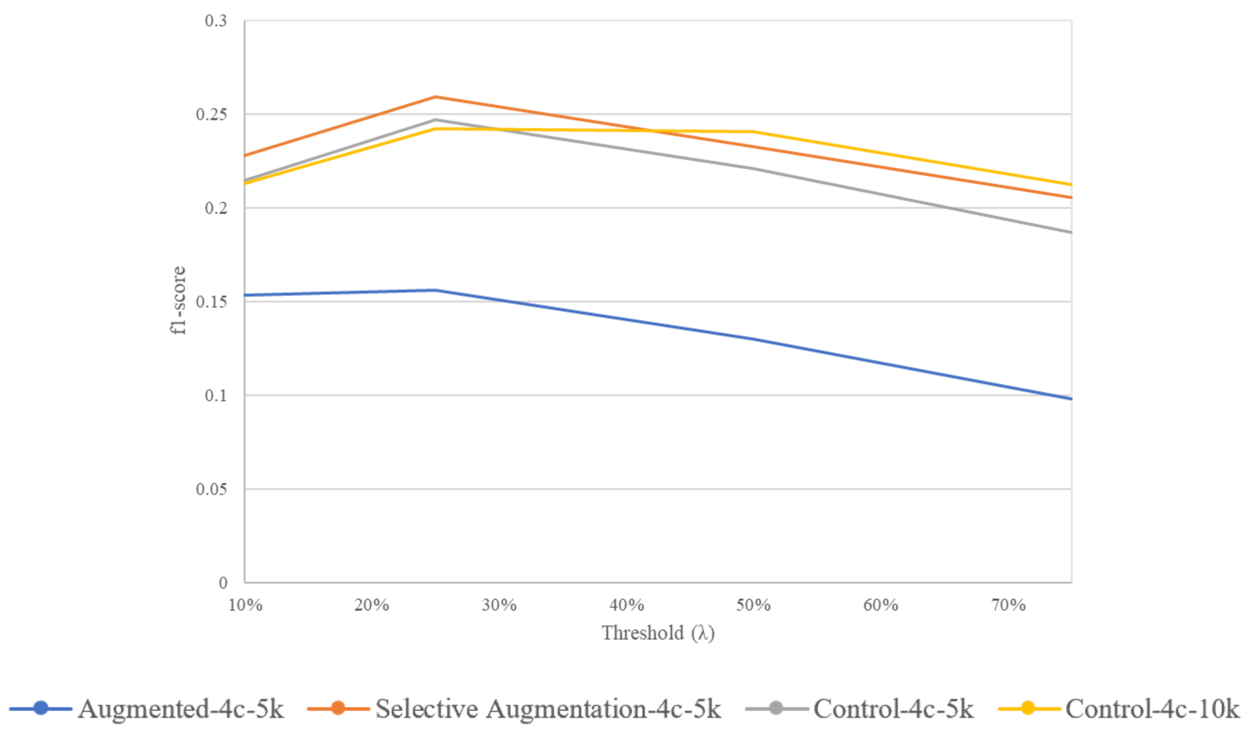


Figure 4.8.6 – IOU 75% Augmentation and Step Evaluation

Table 4.8.1 – F1-Score Difference of Selective Augmentation-4c-5k and Control-4c-5k

	λ^{10}	λ^{25}	λ^{50}	λ^{75}
AP^{35}	-0.016	0.024	0.063	0.082
AP^{50}	0.037	0.024	0.073	0.085
AP^{75}	0.013	0.012	0.011	0.019

Where the negative values or grey-filled cells indicate where the Control-4c-5k had higher f1-score, and the positive or orange-filled cells indicate where the Selective Augmentation-4c-5k had the higher f1-score.

Table 4.8.2 – F1-Score Difference of Control-4c-5k and Control-4c-10k

	λ^{10}	λ^{25}	λ^{50}	λ^{75}
AP^{35}	0.001	-0.043	-0.114	-0.174
AP^{50}	-0.017	-0.025	-0.067	-0.109
AP^{75}	0.001	0.005	-0.019	-0.026
Where the positive values or grey-filled cells indicate where the Control-4c-5k had higher f1-score, and the negative or gold-filled cells indicate where the Control-4c-10k had the higher f1-score.				

CHAPTER 5: DISCUSSION

5.1 Evaluation

A training dataset of 719 unaltered structural bridge detail images were created for detecting *bearings*, *cover plate terminations*, *gusset plate connections*, and *out of plane stiffeners*. The results were evaluated for the dataset overfitting, complete augmentation, selective augmentation, and effect of increasing the number of steps. The effects on the performance of the datasets were compared by examining mAP, precision, accuracy, and f1-scores.

Mean Average Precision

The first set of graphs (Figures 4.2.1 – 4.2.3) evaluated whether the model was overfit to the data. It was hypothesized that the Additional Data-3c-5k would outperform the Control-3c-5k because it was trained on a more diverse set of images. Both curves, Control-3c-5k and Additional Data-3c-5k, generally converged as IOU threshold increase and as the confidence threshold (λ) increased. However, in the IOU 35%, between a 25%-50% (λ), the Control-3c-5k was a stronger model than the predicted model, Additional Data-3c-5k. The fact that the Control-3c-5k performed better is an odd phenomenon and may be attributed to the evaluation dataset. This point is addressed in the error discussion section. Overall, it can be concluded that the model trained on only Norfolk Southern railway bridges, Control-3c-5k, did not overfit, and was able to handle images from different environments. Effectively, this means that the dataset which was annotated is transferable and makes good assumptions.

The next set of graphs (Figures 4.2.4 – 4.2.6) examined augmentation and increasing the number of steps or iterations during training. When evaluating the results, it was important to separate them into two evaluation groups. The first group evaluated the augmentation, which included: Control-4c-5k, Augmented-4c-5k, and Selective Augmentation-4c-5k. The second group evaluated the increase in steps: Control-4c-5k, and Control-4c-10k. For the first group, the complete augmentation, Augmented-4c-5k was the worst preforming, with the Selective Augmenation-4c-5k generally having greater precision than the Control-4c-

5k. For an IOU of 35% and 50%, the selectively augmented dataset was the more precise model by as much as 5.4% (IOU 35%, λ of 50%) (Table 4.2.1). The control was slightly more precise for an IOU of 75%, with the greatest difference 0.7% (λ of 25%). These results indicate that selectively augmenting the data has positive improvements on the precision. For the second evaluation group, increasing the number of step sizes had significant effects on the model precision as the (λ) was raised. At lower (λ) the Control-4c-5k had the stronger performance, but from 25%-75% the Control-4c-10k was the more precise model, with the largest difference in mAP of 12.2% (IOU 35%, λ of 75%). The results of differences between the two models are summarized in Table (4.2.2). The general trend was that as the (λ) increased, the gap between Control-4c-5k and Control-4c-10k widened. The differences from the two models (Control-4c-5k and Control-4c-10k) reiterates that there is room for growth, and other models may achieve significant improvements through a more sophisticated approach to the training configuration parameters (number of steps, etc.). A sample output of predictions for the (C_E) on the Control-4c-5k model was provided in Appendix D.

Precision by Structural Detail

The average percentage drop in precision across all the models was summarized in Table (4.4.1). Table (4.4.1) examined how sensitive each structural detail was to the increase in IOU. *Bearings* and *gusset plate connections* experienced smaller drops in precision from and IOU of 35% to an IOU of 50%. *Bearings* and *gusset plate connections* experienced minimal declines in precision as IOU was increased from 35% to 50% at 8.10% and 4.25% respectfully. *Cover plate terminations* and *out of plane stiffeners* comparatively experienced large percentage declines in precision of 22.54% and 15.73% respectfully. This indicates that *cover plate terminations* and *out of plane stiffeners* were more sensitive to stringent IOU thresholds, and was a commentary on the nature of the bounding boxes on the details. The nature being, for example, that *cover plate terminations* were small, and the bounding box sizes were ambiguous (Figure 4.1.1). On the other hand, *gusset plate connections* and *bearings*, are well defined. These details are squarer, and easier to define than the skinny nature of an *out of plane stiffener* or the ambiguous bounding box size for a *cover* *plate termination* (Figure 4.1.2). Another note of the precision by structural detail is how the models performed on the *gusset plate connection* details. By observation, the spread between these curves are smaller than the rest of the details. The complete augmentation model was able to deliver a more adequate level of precision in relation to the other models. This may be attributed to the simple geometry of a *gusset plate connection*, as well as its symmetric nature.

Average Class Precision

The average class precision measured the average precision of all the structural details. The trends in the average precision show how the precision increases as the confidence threshold (λ) increases. This outcome was expected, as the (λ) rises, the better the prediction - there were fewer predictions, but the predictions were true positives (TP). As the IOU threshold increases, there were losses in precision. This was also expected since the model was confident in an object prediction, but the IOU threshold makes it harder to categorized as a TP. By observation, the overfitting evaluation performed as excepted, with the Additional Data-3c-5k preforming slightly better than the Control-3c-5k. This reaches the same conclusion as made with the mAP, that the model did not overfit. The Selective Augmentation-4c-5k had better precision, overall, than the Control-4c-5k (Table 4.6.1). This matched the conclusions found in the mAP comparison between the two models. By comparison, the selective augmentation had a more convincing impact on the average class precision of the model than on its mAP (Table 4.2.1, 4.6.1). The Control-4c-5k had better accuracy at the (λ) of 25% and 50%, while the Control-4c-10k generally had better accuracy at (λ) of 10% and 75% (Table 4.6.2). It is unclear why there were such clearly defined regions across all IOUs and thresholds. It was expected that the model with more iterations, Control-4c-10k, would perform better across all points.

Accuracy by Structural Detail

The general trend was as the confidence threshold (λ) increased the accuracy decreased. This was expected since the number of objects detected would fall off as the (λ) rises. However, for the bearings, the trend was horizontal with little drop in accuracy as compared to the structural details. This indicates that bearings which were detected had a high confidence. Another general trend was that as the IOU threshold increased, the accuracy decreased. This was expected, however some structural details experienced greater losses in accuracies than others. As in the average precision by structural detail, *cover plate terminations* and *out of plane stiffeners* had the biggest losses in accuracy between IOU thresholds of 35% and 50%. The *cover plate termination* and *out of plane stiffener* losses were both greater than the other details with 26.4% and 21.2% respectfully. This is compared to the *bearing*s and *gusset plate connections* with losses of only 11.5% and 6.1% respectfully. These losses between IOU thresholds are summarized in Table (4.7.1). These trends indicate that *cover plate terminations* and *out of plane stiffeners* were both more sensitive to bounding box predictions. By observation, *gusset plate connections* are the most accurately detected detail, followed by *bearings*. Again, this may be attributed to it simple bounding box geometry.

Although the Augmented-4c-5k preformed the worst in almost every class and threshold case, its curve was curious. It was strange that there was drastic improvement in accuracy performance from the (λ) of 10% to 25%. Some of the other models like Control-4c-10k exhibited minor increases to accuracy (Figure 4.6.1 – Cover Plate Connection). This was due to the fact that there were many false positives (FP) predicted at the low (λ) of 10%. The large amount of FP were reduced after the (λ) was increased. This trend was also seen in the precision by structural detail. This would indicate that while the model was encapsulating more details correctly in the lower (λ), it was also muddling the images with FP. Therefore, a (λ) of 25% is a better balance of accuracy and precision than a (λ) of 10%. Another curiosity of the Augmented-4c-5k model was that for the Gusset Plate Connection IOU of 50% (λ) 25%-50% (Figure 4.6.2), the model had a rare moment of superior accuracy than the Control-4c-10k and Control-4c-5k. This may be attributed to the fact that a gusset plate connection can be oriented in any direction in the real-world, where the other structural details cannot.

Average Class Accuracy

The average class accuracies depict the same general trends seen in the accuracy by structural details. The overfitting evaluation did not change from the previous conclusions made, the model did not overfit.

A summary table for the augmentation evaluation indicated that selectively augmenting the data improved accuracy at all thresholds except at a confidence threshold (λ) of 10% and an IOU 35% (Table 4.7.1). The Selective Augmentation-4c-5k had its greatest difference (5.6%) in accuracy at a (λ) of 50% and an IOU 35%. A summary table for the number of iterations steps indicated that the increased iteration met or improved accuracy at all thresholds (Table 4.7.2). The Control-4c-10k had its greatest difference (12%) in accuracy at a (λ) of 75% and an IOU 35%. This result means that for an IOU of 35%, the Control-4c-10k had a significantly larger amount of highly confident predictions.

F1-Scores

The f1-scores were used to define which model was best when both precision and accuracy was considered. For overfitting the conclusions remained constant by observation. For complete augmentation, it was clear from observation that the model did not do as well as its peers. For the selective augmentation evaluation, it was clear from Table (4.8.1) that the Selective Augmentation-4c-5k was the better performing model when accuracy and precision were considered. As the confidence thresholds (λ) increased, the f1-score difference between the models increased. This trend indicates that as the λ increased, selectively augmenting the data has a greater impact on the output. For the step evaluation, the Control-4c-10k had the higher f1-scores as the λ increased. In general, the Control-4c-10k either matched or exceed the Control-4c-5k (Table 4.8.2). This trend indicates that increasing the number of iteration steps will improve accuracy and precision and will become more important as λ increases.

In considering the accuracy results and the f1-scores, it was recommended that the optimal operation is at a (λ) of 25%. By observation, the f1-scores showed little decline from a 10% to a 25% (λ) on IOUs of 35% and 50%. For the IOU of 75% f1-score graph, the model peaks at 25%. Consider the Selective Augmentation-4c-5k model at an IOU of 50% (Figure 4.5.5). As demonstrated by the decline in the f1-score graphs, the precision gains from 25% - 75% (Figure 4.5.5) did not compare to the losses in its accuracy (Figure 4.7.5) after increasing in λ. Furthermore, it was stated that too many FP would be distracting to an inspector. The higher λ for prediction reduces the potential for congestion of the FP that would distract and impede an inspector's progress.

5.2 Error

The main sources of error were from how the dataset was annotated, how the data was split, and possibly from how the models were trained. There were not clearly defined rules on how each structural detail should be annotated. For example, in Figure (5.2.1), there are *bearings* boxed in yellow, which are far away, that were not annotated since they were so small in the frame. However, these types of decisions would have affected how the model preformed. It was also noted that some details were being correctly predicted, in the evaluation images, that were not considered when drawing ground truth boxes. The reason that these structural details were not originally being considered were because on small portions of the structural detail was visible (only the corners of them were showing, etc.). By not annotating these minuscule slivers of the details it would have confused the model since these were correct predictions. After viewing the results of some initial runs on the structural details, the evaluation dataset was adjusted to encapsulate some of these correct predictions on slivers of details. Another source of error came from the inevitability that some details were simply missed. Figure (5.2.1) has a red box around a *gusset plate connection* that was missed during the annotation process.

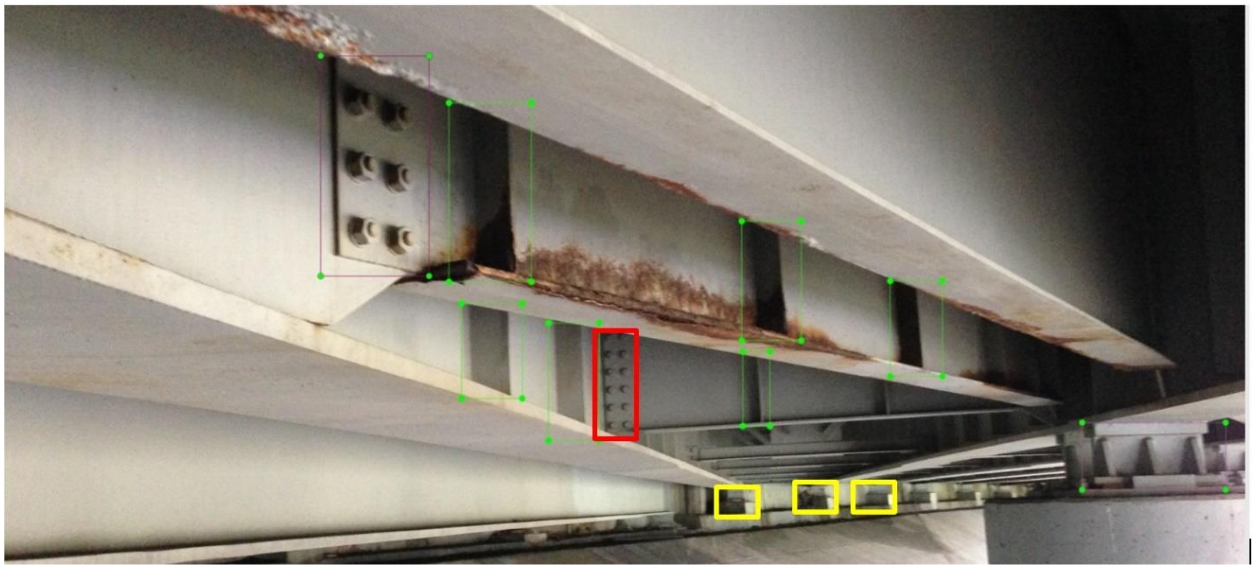
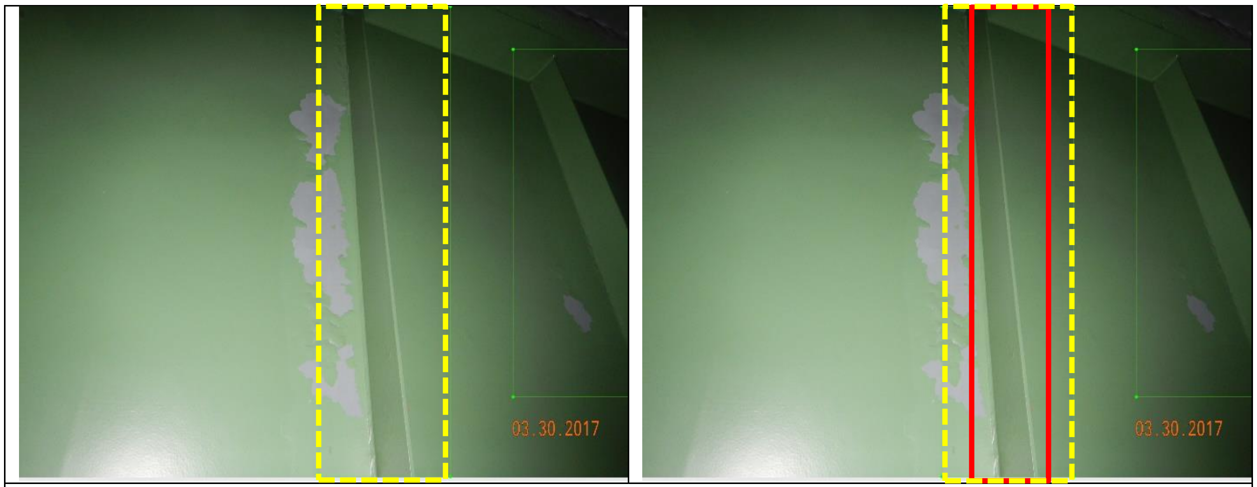


Figure 5.2.1 - Missed Gusset Plate Connection (Red), Far Away Bearings (Yellow)

The dataset was annotated in sprints. The first annotation sprint included Norfolk Southern images. The second sprint was many weeks later and was on the Clark Nexsen images. Since there were not defined rules on quality control, the consistency of the bounding boxes may have changed from week to week. It would have been useful to define these levels of quality for each structural detail. For example, in Figure (5.2.2), the bounding box around the *out of plane stiffener* could have been drawn more precisely. The dashed yellow line indicates what was actually drawn, where the solid line red box indicates what could have been the more precise encapsulation of the structural detail.



Yellow dashed line is what was drawn in the dataset. The red solid line is the ideal bounding box.

Figure 5.2.2 - Bounding Box Inconsistencies

As mentioned in the methodology section, the dataset split was not ideal. It was not ideal since there were disparities among the percentage of structural details in the evaluation dataset for each class. For example, *gusset plate connections* had 15.77% object instances in the evaluation dataset, while *bearings* only had 4.7% (Table 3.5.12). To have a more balanced evaluation dataset, it would have been best to have a more even percentage of structural detail instances. By keeping the percentage equal across classes, it makes for a better comparison of class performance, and a more reliable mAP. Additionally, it was conceived that there could have been a stronger way to compare the model overfitting performance between the Control-3c-5k and the Additional Data-3c-5k. The Control-3c-5k was made up of only Norfolk Southern railroad box-girder bridges, while the Additional Data-3c-5k comprised of Norfolk Southern railroad box-girder bridges as well as many other bridge types from Clark Nexsen. The evaluation dataset used images from both Clark Nexsen and Norflok Southern. It was conceived that a stronger evaluation may be to use only Clark Nexsen images in order to evaluate the performance of only the un-seen images rather than mix the two sources.

There was not much focus that went into refining the training of the models. For example, the batch size was arbitrarily chosen, and other parameters such as the learning rate was not adjusted. By not adjusting these parameters, it may have affected the performance of the different tests. For example, the *complete* *augmentation* may have required more training steps since were more eight times as many photos as the control. With the increased number of training steps perhaps the model would have behaved differently and outperformed the other models. Although this level of test optimization was outside the scope of the research objectives, it should be noted that it has the potential for altering the findings in this experiment. A more robust experiment might consider more sophisticated model training practices for a more robust conclusion.

CHAPTER 6: CONCLUSION

This paper introduced COCO-Bridge, an annotated structural detail dataset for unmanned aircraft systems (UAS)-assisted bridge inspections. Recent research has focused much effort towards structural *defect* identification, whereas COCO-Bridge is a dataset for structural *detail* identification. Through the identification of structural details, the artificial intelligence, the UAS, and the inspector have greater context to make decisions regarding the defects detected.

There were 774 images annotated, identifying four structural detail classes: *bearings*, *cover plate terminations*, *gusset plate connections*, and *out of plane stiffeners*. Convolutional neural networks (CNN) were trained on several datasets designed to understand the nature of the target structural details. The nature of these structural details was determined through dataset comparisons to evaluate model overfitting, augmentation, and model optimization.

A process for gathering, annotating, training, and testing the CNN was modeled in the paper. During the data gathering process, photos were self-taken and collected from engineering firms and departments of transportation. The greatest success was found using the images provided by collaborating engineering firms. It was found that sorting the data based on the fabrication material (i.e steel, wood, concrete) was the most economical method images from engineering firms. Upon reflection of the annotation process, it was determined that a standard annotation guideline for structural details needed to be established for consistency. Additionally, it was found that mirroring the images about the vertical axis improved both precision and accuracy. For training, the model showed significant improvement by increasing the number of computational iterations. The operating confidence threshold of the model was found to be 25%, as it provided a reasonable balance of both accuracy and precision to the model. A successful proof of concept of an in-field UAS test was outlined to be modeled in the field for future experimentation. As a result of these findings, general and future growth recommendations have been identified. This dataset and paper is the first step in building COCO-Bridge for the expansion of artificial intelligence in the structural inspection space.

6.1Recommendations

Based on the research herein, the following procedures are recommended while developing a more robust and extensive annotated image dataset, and CNN model configurations:

Data Gathering

- Bridges fabricated from different materials (e.g. steel, concrete, timber, etc.) should have their own separate datasets and neural network models trained independently from one another. CNN models perform better on a more focused area of training, rather than training models on a vastly broad scope of classes. Therefore, compartmentalizing datasets by bridge material type would be more ideal for the CNN models.
- Bridge construction types (e.g. plate girder, truss, tied-arch, cable stayed, suspension, etc.) have significantly different construction details and should therefore be trained with their datasets and neural network models trained independently from one another. As previously mentioned, CNN models perform better on a more focused area of training.

Dataset Annotation

- Quality control rules with clearly defined requirements should be established for annotating images to ensure high quality training images with consistency throughout the dataset. Having quality control rules will minimize inconsistencies and missed structural details in annotations.
- Multiple annotations rounds are recommended on each dataset. Each round is when a human reviews the annotations for correctness, completeness, and accuracy. It was found that second and third rounds over an annotated dataset helped to eliminate missing structural details and allowed the annotator to focus more on quality control. Therefore, it is recommended to have at least two passes over a dataset, focusing on the following:
  - The first round of dataset annotation should attempt to capture all structural details to the acceptable limits of the quality control rules.

    - The second round, which is inherently quicker, should focus on the refinement of the first round to ensure dataset quality.
- An open-sourced annotation tool that fits the needs based on the output file format and annotation style is recommended to prevent duplication of effort. If the output file desired is XML and the annotation style desired is a bounding box, then LabelImg is the recommended annotation tool to complement the current effort.

Dataset Augmentation

- Images should be re-scaled to a size that can be efficiently handled by the training GPU. For current computational limits, it is recommended to limit image size to 400 x 400 pixels.
- Images may be selectively augmented to maximize the efficiency of available images. Images should only be mirrored about the vertical-axis to achieve increased precision and accuracy without requiring additional images.

Dataset Split

- The dataset should be divided into approximately 15-20% for evaluation images and 80-85% for training images based upon the standard practice in the machine learning/computer vision field (Cordts et al., 2016).
- The fraction of object instances used for evaluation and training should be consistent across all classes to ensure a fair representation of data. For example, if there were 1000 object instances of *gusset plate connections*, and 500 *bearings*, and the data was split 20% evaluation – 80% training, then there should be 200 gusset plate and 100 bearing object instances used in the evaluation dataset to ensure each detail is equally represented.

Training and Testing

- A CNN model that meets the needs of its required task should be selected by examining its speed and mAP metrics. The mAP metrics against a benchmark dataset typically come with the CNN model, which is how developers compare performance.

- The number of steps in a model should be increased until the enhancements to precision and accuracy are negligible. In this research paper, models were not trained until their performance was negligible between models. For example, the Control-4c-10k model showed significant improvement over the Control-4c-5k model. Therefore, the number of steps should be increased until the performance metrics (mAP, accuracy, etc.) converge.
- For practice based models, the optimal operation was found to be a confidence threshold (λ) of 25%.

6.2 Future Experimentation

Adding Data to the COCO-Bridge Structural Detail Dataset

Data is usually the limiting factor in a model's room for improvement. Additional annotated images gathered from sources or made synthetically should be added to the four selected details within the dataset, It is perceived that structural details may be able to be synthetically generated by building them in a program such as Revitt and super-imposing them into images. More annotated data will improve precision, accuracy, and its ability to recognize unfamiliar images. Also, additional structural details should also be considered, making the model more robust and useful in a broader set of applications. The research community would benefit from the development of an open-source, annotated, structural detail dataset. Researchers would have access to download the data for their own use, or contribute their own data for the expansion of COCO-Bridge.

These additional details that are added to the dataset will have to be annotated and incorporated into the existing dataset as part of the expansion process. Annotation is a time-extensive process, therefore techniques for expediting the process should be explored. For example, using the structural detail output predictions from a trained model on new, un-annotated data is an annotation technique which could be explored to expedite the annotation effort. This technique would effectively have the model do some preliminary predictions to save time on annotating all of the structural details from scratch.

Quality Control Measures for Annotating Structural Bridge Details

Defining specific quality control measures is essential for the continual development of COCO-Bridge. These quality control measures should include guidelines on the scope of the annotations in terms of capturing far-away or cut-off structural details, and acceptable ranges for bounding boxes surrounding structural details. Without having quality control measures, the annotated data from other contributors may result in highly inconsistent annotations. A list of best-practices for each type of structural detail should be defined to deliver consistent annotations provided by users in an open-source environment.

Testing the Existing Dataset with Quality Control Alterations

Evaluation of the existing dataset with differing levels of quality control will be valuable to see model performance shift. Additionally, re-testing the model on a more balanced evaluation dataset would be valuable for a more reliable mAP, structural detail average precision, and accuracy representation. Furthermore, re-testing the dataset by evaluating different training configurations and superior model types would be useful for evaluating the best in-field testing. This research did not examine the effects of training configurations on model performance for augmentation, selective augmentation, etc. This research also did not consider comparing alternative model types such as YOLO (You Only Look Once) (Redmon et al., 2015) when evaluating the model performance. Lastly, the run-time training code could be optimized by altering the script to be multi-threaded, to be shared across multiple GPUs.

Contextualized Defect Identification

The fusion of defect detection on top of detail identification would add another level of complexity to the model and enhance what the model is capable of accomplishing. This would enable the model to identify only defects that were found inside structural detail bounding box regions. This would improve the model's output by eliminating the noise from regions which are not focused on during inspection, as well as bring context to the defects. For example, instead of detecting a crack, rust, spalling, etc., the model could detect that there was a crack *within* a specific structural detail.

Augmented Reality

The annotated structural detail dataset provides an opportunity for augmented reality applications. Augmented reality is when a technology superimposes a computer-generated image onto the field of view of a person using a viewing device. Given the correct identification of the structural detail, it could cue the target regions which are susceptible to defects, such as fatigue cracking.

Applying the Model

Further investigation is need to demonstrate the application of the model in the field. Beyond just creating a dataset, the inspiration for the research was to assist UAS in bridge inspections. Testing the model in the field to evaluate its feasibility, limitations, and inspection enhancements would be valuable for the progression of UAS assisted inspection. An additional application of field testing could be the potential for flight-planning and autonomous navigation from the visual cues that a UAS receives from the model predictions.

APPENDIX A - PROGRAM ARCHITECTURE

Nine models were presented in the report, each requiring its own pre-processing and post-processing directory to store its referenced data, model-specific folders, and result output. To mitigate errors and maintain consistency across models, most of the helper-scripts were centralized to one directory. Centralization effectively reduces the copies of the same files. Instead of many duplicates of the same files in each model directory, there was one main file which would call the helper scripts. A many to one relationship require more up-front effort and introduce additional complexity, however, it reaps benefits by being easy to debug and update. The program architecture, python scripts, and supporting files may be found on the github site[8].

[8] https://github.com/beric7/COCO-Bridge

APPENDIX B – RESULTS DATA

"toy" model					
IOU-λ		AP	TP	FP	mAP
35-10%	Cover Plate Termination	0.0%	0	0	53.22%
	Gusset Plate Connection	98.3%	22	3	
	Out of Plane Stiffener	61.4%	8	5	
35-25%	Cover Plate Termination	0.0%	0	0	47.25%
	Gusset Plate Connection	85.9%	19	2	
	Out of Plane Stiffener	55.8%	7	3	
35-50%	Cover Plate Termination	0.0%	0	0	43.94%
	Gusset Plate Connection	81.8%	18	1	
	Out of Plane Stiffener	50.0%	6	1	
35-75%	Cover Plate Termination	0.0%	0	0	26.77%
	Gusset Plate Connection	63.6%	14	0	
	Out of Plane Stiffener	16.7%	2	0	
50-10%	Cover Plate Termination	0.0%	0	0	51.04%
	Gusset Plate Connection	98.3%	22	3	
	Out of Plane Stiffener	54.9%	7	6	
50-25%	Cover Plate Termination	0.0%	0	0	45.31%
	Gusset Plate Connection	85.9%	19	2	
	Out of Plane Stiffener	50.0%	6	4	
50-50%	Cover Plate Termination	0.0%	0	0	43.94%
	Gusset Plate Connection	81.8%	18	1	
	Out of Plane Stiffener	50.0%	6	1	
50-75%	Cover Plate Termination	0.0%	0	0	26.77%
	Gusset Plate Connection	63.6%	14	0	
	Out of Plane Stiffener	16.7%	2	0	
75-10%	Cover Plate Termination	0.0%	0	0	19.77%
	Gusset Plate Connection	53.8%	14	11	
	Out of Plane Stiffener	5.6%	2	11	
75-25%	Cover Plate Termination	0.0%	0	0	19.77%
	Gusset Plate Connection	53.8%	14	7	
	Out of Plane Stiffener	5.6%	2	8	
75-50%	Cover Plate Termination	0.0%	0	0	18.76%
	Gusset Plate Connection	50.7%	13	6	
	Out of Plane Stiffener	5.6%	2	5	
75-75%	Cover Plate Termination	0.0%	0	0	14.36%
	Gusset Plate Connection	43.1%	11	3	
	Out of Plane Stiffener	0.0%	0	2	
GT	Cover Plate Termination	2			
	Gusset Plate Connection	22			
	Out of Plane Stiffener	13			

Figure B.1 – "Toy" model

Control-3c-5k					
IOU-λ		**AP**	**TP**	**FP**	**mAP**
35-10%	Cover Plate Termination	62.04%	8	4	52.82%
	Gusset Plate Connection	62.88%	49	9	
	Out of Plane Stiffener	33.53%	31	18	
35-25%	Cover Plate Termination	54.17%	7	1	46.06%
	Gusset Plate Connection	55.84%	43	5	
	Out of Plane Stiffener	28.18%	25	7	
35-50%	Cover Plate Termination	46.43%	6	1	37.37%
	Gusset Plate Connection	44.84%	34	3	
	Out of Plane Stiffener	20.84%	18	4	
35-75%	Cover Plate Termination	16.67%	2	0	19.66%
	Gusset Plate Connection	35.71%	27	1	
	Out of Plane Stiffener	6.61%	6	1	
50-10%	Cover Plate Termination	14.81%	4	8	33.33%
	Gusset Plate Connection	59.46%	47	11	
	Out of Plane Stiffener	25.73%	26	23	
50-25%	Cover Plate Termination	10.71%	3	5	28.48%
	Gusset Plate Connection	52.72%	41	7	
	Out of Plane Stiffener	22.02%	21	11	
50-50%	Cover Plate Termination	10.71%	3	4	23.82%
	Gusset Plate Connection	43.34%	33	4	
	Out of Plane Stiffener	17.41%	16	6	
50-75%	Cover Plate Termination	0.00%	0	2	13.51%
	Gusset Plate Connection	35.71%	27	1	
	Out of Plane Stiffener	4.82%	5	2	
75-10%	Cover Plate Termination	0.00%	0	12	11.83%
	Gusset Plate Connection	33.67%	29	29	
	Out of Plane Stiffener	1.81%	7	42	
75-25%	Cover Plate Termination	0.00%	0	8	11.74%
	Gusset Plate Connection	33.67%	29	19	
	Out of Plane Stiffener	1.56%	6	26	
75-50%	Cover Plate Termination	0.00%	0	7	10.87%
	Gusset Plate Connection	31.79%	27	10	
	Out of Plane Stiffener	0.81%	3	19	
75-75%	Cover Plate Termination	0.00%	0	2	9.22%
	Gusset Plate Connection	27.35%	23	5	
	Out of Plane Stiffener	0.31%	1	6	
GT	Cover Plate Termination	12			
	Gusset Plate Connection	75			
	Out of Plane Stiffener	80			

Figure B.2 – Control-3c-5k

Additional Data-3c-5k					
IOU-λ		**AP**	**TP**	**FP**	**mAP**
35-10%	Cover Plate Termination	66.67%	8	1	61.04%
	Gusset Plate Connection	71.31%	55	17	
	Out of Plane Stiffener	45.13%	41	22	
35-25%	Cover Plate Termination	33.33%	4	0	41.23%
	Gusset Plate Connection	63.49%	48	4	
	Out of Plane Stiffener	26.88%	23	4	
35-50%	Cover Plate Termination	25.00%	3	0	30.97%
	Gusset Plate Connection	55.75%	42	1	
	Out of Plane Stiffener	12.16%	10	1	
35-75%	Cover Plate Termination	16.67%	2	0	22.35%
	Gusset Plate Connection	46.63%	35	1	
	Out of Plane Stiffener	3.75%	3	0	
50-10%	Cover Plate Termination	27.43%	5	4	40.12%
	Gusset Plate Connection	64.55%	50	22	
	Out of Plane Stiffener	28.39%	33	30	
50-25%	Cover Plate Termination	8.33%	2	2	28.44%
	Gusset Plate Connection	60.50%	46	6	
	Out of Plane Stiffener	16.48%	18	9	
50-50%	Cover Plate Termination	2.78%	1	2	21.57%
	Gusset Plate Connection	54.28%	41	2	
	Out of Plane Stiffener	7.64%	8	3	
50-75%	Cover Plate Termination	0.00%	0	2	16.10%
	Gusset Plate Connection	46.63%	35	1	
	Out of Plane Stiffener	1.67%	2	1	
75-10%	Cover Plate Termination	6.67%	2	7	14.79%
	Gusset Plate Connection	34.29%	33	39	
	Out of Plane Stiffener	3.40%	9	54	
75-25%	Cover Plate Termination	2.08%	1	3	12.45%
	Gusset Plate Connection	32.32%	30	22	
	Out of Plane Stiffener	2.96%	7	20	
75-50%	Cover Plate Termination	0.00%	0	3	10.49%
	Gusset Plate Connection	29.96%	27	16	
	Out of Plane Stiffener	1.50%	3	8	
75-75%	Cover Plate Termination	0.00%	0	2	9.22%
	Gusset Plate Connection	27.35%	23	5	
	Out of Plane Stiffener	0.31%	1	6	
GT	Cover Plate Termination	12			
	Gusset Plate Connection	75			
	Out of Plane Stiffener	80			

Figure 6.2.1 – Additional Data-3c-5k

Augmented-4c-5k					
IOU-λ		**AP**	**TP**	**FP**	**mAP**
35-10%	Bearing	39.87%	12	11	44.29%
	Cover Plate Termination	37.30%	7	20	
	Gusset Plate Connection	62.78%	70	147	
	Out of Plane Stiffener	37.21%	69	113	
35-25%	Bearing	35.35%	10	1	34.15%
	Cover Plate Termination	28.57%	4	0	
	Gusset Plate Connection	53.22%	54	23	
	Out of Plane Stiffener	19.47%	28	19	
35-50%	Bearing	28.40%	8	1	17.62%
	Cover Plate Termination	0.00%	0	0	
	Gusset Plate Connection	35.16%	34	7	
	Out of Plane Stiffener	6.90%	8	0	
35-75%	Bearing	18.52%	5	0	10.96%
	Cover Plate Termination	0.00%	0	0	
	Gusset Plate Connection	23.60%	22	2	
	Out of Plane Stiffener	1.72%	2	0	
50-10%	Bearing	29.65%	9	14	34.54%
	Cover Plate Termination	24.32%	6	21	
	Gusset Plate Connection	58.89%	66	151	
	Out of Plane Stiffener	25.28%	53	129	
50-25%	Bearing	27.69%	8	3	27.99%
	Cover Plate Termination	17.86%	3	1	
	Gusset Plate Connection	51.04%	52	25	
	Out of Plane Stiffener	15.35%	22	25	
50-50%	Bearing	25.00%	7	2	16.77%
	Cover Plate Termination	0.00%	0	0	
	Gusset Plate Connection	35.16%	34	7	
	Out of Plane Stiffener	6.90%	8	0	
50-75%	Bearing	18.52%	5	0	10.96%
	Cover Plate Termination	0.00%	0	0	
	Gusset Plate Connection	23.60%	22	2	
	Out of Plane Stiffener	1.72%	2	0	

Figure B.4 – Augmented-4c-5k

Augmented-4c-5k					
IOU-λ		**AP**	**TP**	**FP**	**mAP**
75-10%	Bearing	8.28%	4	19	9.43%
	Cover Plate Termination	1.73%	2	25	
	Gusset Plate Connection	25.94%	36	181	
	Out of Plane Stiffener	1.76%	14	168	
75-25%	Bearing	7.14%	3	8	7.98%
	Cover Plate Termination	0.00%	0	4	
	Gusset Plate Connection	23.83%	31	46	
	Out of Plane Stiffener	0.95%	6	41	
75-50%	Bearing	7.14%	3	6	6.28%
	Cover Plate Termination	0.00%	0	0	
	Gusset Plate Connection	17.82%	21	20	
	Out of Plane Stiffener	0.17%	1	7	
75-75%	Bearing	5.56%	2	3	4.90%
	Cover Plate Termination	0.00%	0	0	
	Gusset Plate Connection	14.03%	15	9	
	Out of Plane Stiffener	0.00%	0	2	
GT	Bearing	27			
	Cover Plate Termination	14			
	Gusset Plate Connection	89			
	Out of Plane Stiffener	116			

Figure B.4 – Augmented-4c-5k

Selective Augmentation-4c-5k					
IOU-λ		**AP**	**TP**	**FP**	**mAP**
35-10%	Bearing	46.54%	13	4	56.14%
	Cover Plate Termination	54.29%	8	4	
	Gusset Plate Connection	70.65%	66	37	
	Out of Plane Stiffener	53.06%	70	45	
35-25%	Bearing	40.74%	11	0	45.04%
	Cover Plate Termination	42.86%	6	0	
	Gusset Plate Connection	57.41%	52	3	
	Out of Plane Stiffener	39.13%	48	9	
35-50%	Bearing	33.33%	9	0	32.60%
	Cover Plate Termination	28.57%	4	0	
	Gusset Plate Connection	45.53%	41	1	
	Out of Plane Stiffener	22.97%	27	2	
35-75%	Bearing	33.33%	9	0	21.17%
	Cover Plate Termination	7.14%	1	0	
	Gusset Plate Connection	27.92%	25	1	
	Out of Plane Stiffener	16.29%	19	1	
50-10%	Bearing	42.37%	12	5	46.69%
	Cover Plate Termination	49.18%	8	4	
	Gusset Plate Connection	61.11%	59	44	
	Out of Plane Stiffener	34.09%	53	62	
50-25%	Bearing	37.04%	10	1	36.42%
	Cover Plate Termination	30.95%	5	1	
	Gusset Plate Connection	51.84%	48	7	
	Out of Plane Stiffener	25.85%	35	22	
50-50%	Bearing	33.33%	9	0	28.27%
	Cover Plate Termination	17.86%	3	1	
	Gusset Plate Connection	42.68%	39	3	
	Out of Plane Stiffener	19.22%	24	5	
50-75%	Bearing	33.33%	9	0	20.49%
	Cover Plate Termination	7.14%	1	0	
	Gusset Plate Connection	26.79%	24	2	
	Out of Plane Stiffener	14.69%	18	2	

Figure B.5 – Selective Augmentation-4c-5k

Selective Augmentation-4c-5k					
IOU-λ		**AP**	**TP**	**FP**	**mAP**
50-75%	Bearing	33.33%	9	0	20.49%
	Cover Plate Termination	7.14%	1	0	
	Gusset Plate Connection	26.79%	24	2	
	Out of Plane Stiffener	14.69%	18	2	
75-10%	Bearing	23.05%	7	10	14.27%
	Cover Plate Termination	1.19%	1	11	
	Gusset Plate Connection	25.29%	33	70	
	Out of Plane Stiffener	7.56%	18	97	
75-25%	Bearing	23.05%	7	4	13.35%
	Cover Plate Termination	1.19%	1	5	
	Gusset Plate Connection	22.04%	27	28	
	Out of Plane Stiffener	7.10%	16	41	
75-50%	Bearing	23.05%	7	2	12.19%
	Cover Plate Termination	0.00%	0	4	
	Gusset Plate Connection	19.56%	23	19	
	Out of Plane Stiffener	6.16%	13	16	
75-75%	Bearing	23.05%	7	2	10.72%
	Cover Plate Termination	0.00%	0	1	
	Gusset Plate Connection	14.99%	16	10	
	Out of Plane Stiffener	4.83%	10	10	
GT	Bearing	27			
	Cover Plate Termination	14			
	Gusset Plate Connection	89			
	Out of Plane Stiffener	116			

Figure B.5 - Selective Augmentation-4c-5k

Control-4c-5k					
IOU-λ		**AP**	**TP**	**FP**	**mAP**
35-10%	Bearing	62.79%	19	12	61.19%
	Cover Plate Termination	55.56%	8	5	
	Gusset Plate Connection	69.27%	66	37	
	Out of Plane Stiffener	57.12%	76	40	
35-25%	Bearing	43.52%	12	4	44.10%
	Cover Plate Termination	42.86%	6	0	
	Gusset Plate Connection	56.96%	52	8	
	Out of Plane Stiffener	33.04%	41	8	
35-50%	Bearing	37.04%	10	0	27.25%
	Cover Plate Termination	14.29%	2	0	
	Gusset Plate Connection	45.60%	41	3	
	Out of Plane Stiffener	12.07%	14	0	
35-75%	Bearing	29.63%	8	0	16.22%
	Cover Plate Termination	0.00%	0	0	
	Gusset Plate Connection	29.21%	26	0	
	Out of Plane Stiffener	6.03%	7	0	
50-10%	Bearing	44.56%	13	18	44.10%
	Cover Plate Termination	32.74%	7	6	
	Gusset Plate Connection	62.59%	60	43	
	Out of Plane Stiffener	36.49%	60	56	
50-25%	Bearing	39.58%	11	5	33.88%
	Cover Plate Termination	19.05%	4	2	
	Gusset Plate Connection	54.43%	50	10	
	Out of Plane Stiffener	22.44%	34	15	
50-50%	Bearing	37.04%	10	0	22.97%
	Cover Plate Termination	3.57%	1	1	
	Gusset Plate Connection	44.31%	40	4	
	Out of Plane Stiffener	6.94%	10	4	
50-75%	Bearing	29.63%	8	0	15.61%
	Cover Plate Termination	0.00%	0	0	
	Gusset Plate Connection	29.21%	26	0	
	Out of Plane Stiffener	3.60%	5	2	

Figure B.6 – Control-4c-5k

Control-4c-5k					
IOU-λ		**AP**	**TP**	**FP**	**mAP**
75-10%	Bearing	24.51%	8	23	14.85%
	Cover Plate Termination	3.17%	2	11	
	Gusset Plate Connection	25.38%	29	74	
	Out of Plane Stiffener	6.32%	15	101	
75-25%	Bearing	24.51%	8	8	14.04%
	Cover Plate Termination	1.19%	1	5	
	Gusset Plate Connection	24.60%	27	33	
	Out of Plane Stiffener	5.84%	12	37	
75-50%	Bearing	22.65%	7	3	12.43%
	Cover Plate Termination	0.00%	0	2	
	Gusset Plate Connection	22.25%	23	21	
	Out of Plane Stiffener	4.81%	8	6	
75-75%	Bearing	20.06%	6	2	10.30%
	Cover Plate Termination	0.00%	0	0	
	Gusset Plate Connection	18.79%	18	8	
	Out of Plane Stiffener	2.34%	4	3	
GT	Bearing	27			
	Cover Plate Termination	14			
	Gusset Plate Connection	89			
	Out of Plane Stiffener	116			

Figure B.6 – Control-4c-5k

Control-4c-10k					
IOU-λ		AP	TP	FP	mAP
35-10%	Bearing	55.03%	16	7	55.95%
	Cover Plate Termination	48.21%	7	3	
	Gusset Plate Connection	67.11%	61	14	
	Out of Plane Stiffener	53.43%	71	44	
35-25%	Bearing	46.75%	13	4	49.18%
	Cover Plate Termination	48.21%	7	1	
	Gusset Plate Connection	60.90%	55	3	
	Out of Plane Stiffener	40.86%	51	12	
35-50%	Bearing	40.74%	11	2	37.73%
	Cover Plate Termination	35.71%	5	0	
	Gusset Plate Connection	45.45%	41	1	
	Out of Plane Stiffener	29.00%	35	3	
35-75%	Bearing	40.74%	11	1	28.41%
	Cover Plate Termination	21.43%	3	0	
	Gusset Plate Connection	33.27%	30	1	
	Out of Plane Stiffener	18.20%	22	2	
50-10%	Bearing	47.87%	14	9	42.09%
	Cover Plate Termination	22.32%	5	5	
	Gusset Plate Connection	62.01%	57	18	
	Out of Plane Stiffener	36.16%	58	57	
50-25%	Bearing	40.74%	11	6	37.01%
	Cover Plate Termination	22.32%	5	3	
	Gusset Plate Connection	57.98%	53	5	
	Out of Plane Stiffener	27.00%	39	24	
50-50%	Bearing	40.74%	11	2	28.24%
	Cover Plate Termination	7.14%	2	3	
	Gusset Plate Connection	43.89%	40	2	
	Out of Plane Stiffener	21.17%	29	9	
50-75%	Bearing	40.74%	11	1	22.33%
	Cover Plate Termination	2.38%	1	2	
	Gusset Plate Connection	32.08%	29	2	
	Out of Plane Stiffener	14.10%	19	5	

Figure B.7 – Control-4c-10k

Control-4c-10k					
IOU-λ		**AP**	**TP**	**FP**	**mAP**
75-10%	Bearing	20.99%	7	16	14.82%
	Cover Plate Termination	0.00%	0	10	
	Gusset Plate Connection	30.83%	34	41	
	Out of Plane Stiffener	7.46%	18	97	
75-25%	Bearing	20.99%	7	10	14.66%
	Cover Plate Termination	0.00%	0	8	
	Gusset Plate Connection	30.20%	33	25	
	Out of Plane Stiffener	7.46%	18	45	
75-50%	Bearing	20.99%	7	6	13.38%
	Cover Plate Termination	0.00%	0	5	
	Gusset Plate Connection	25.98%	27	15	
	Out of Plane Stiffener	6.56%	15	23	
75-75%	Bearing	20.99%	7	5	11.67%
	Cover Plate Termination	0.00%	0	3	
	Gusset Plate Connection	22.03%	22	9	
	Out of Plane Stiffener	3.65%	8	16	
GT	Bearing	27			
	Cover Plate Termination	14			
	Gusset Plate Connection	89			
	Out of Plane Stiffener	116			

Figure B.8 – Control-4c-10k

APPENDIX C – EVALUATION DATASET (C_T)

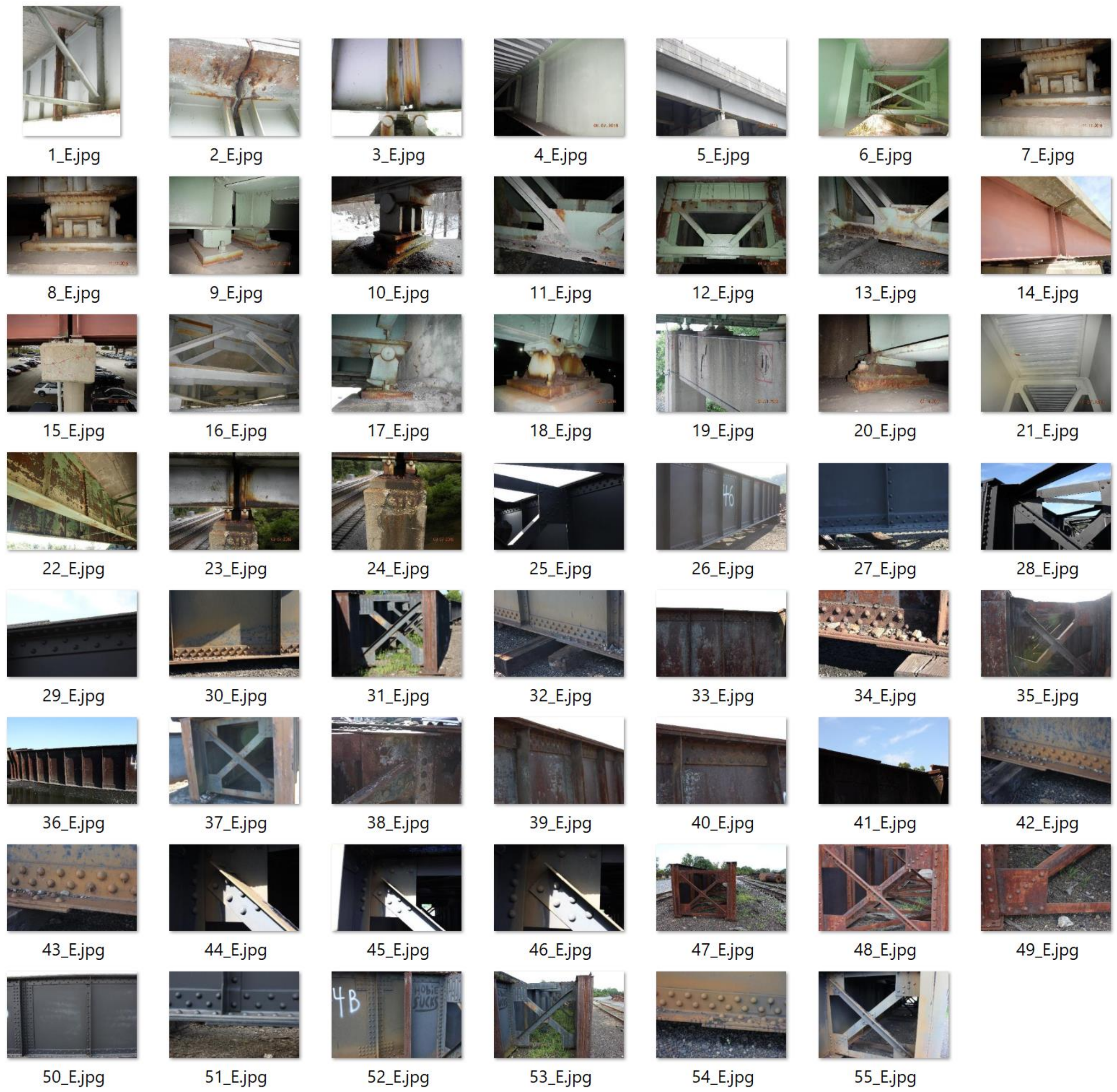


Figure C.1 – Evaluation Dataset for C_T

APPENDIX D – CONTROL-4C-5K (C_T) Λ-25%, IOU-50%

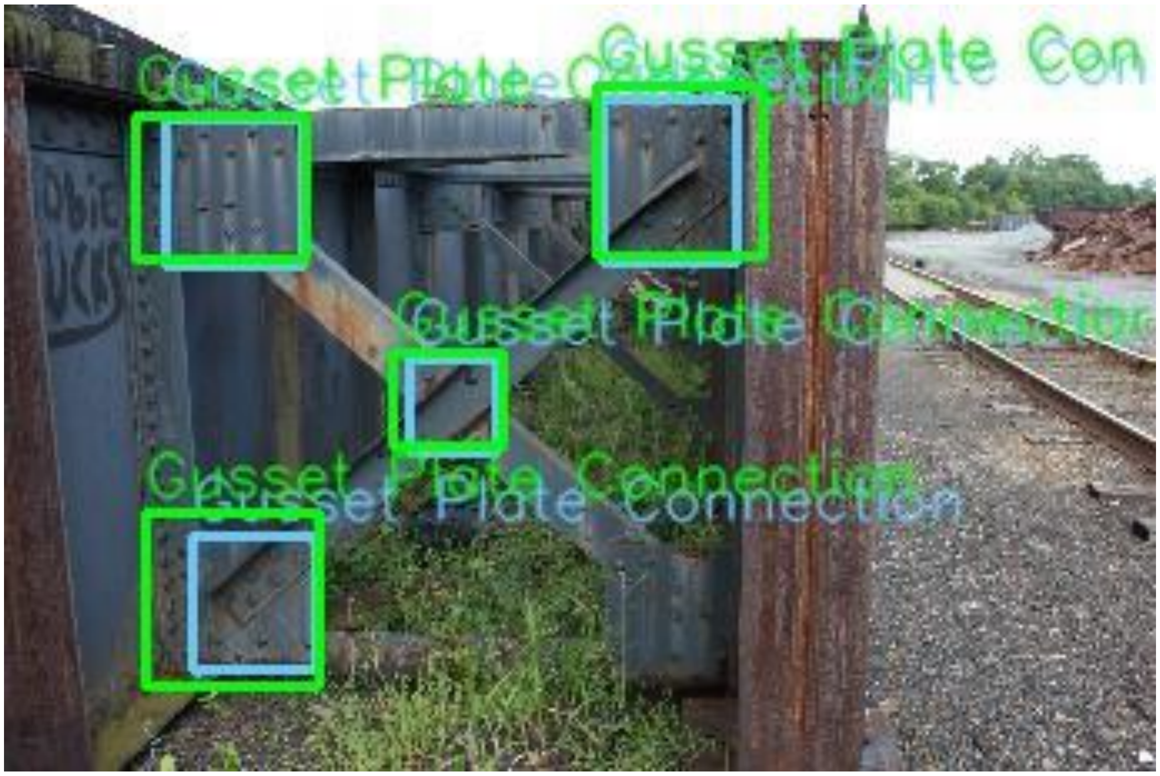


Figure D.1

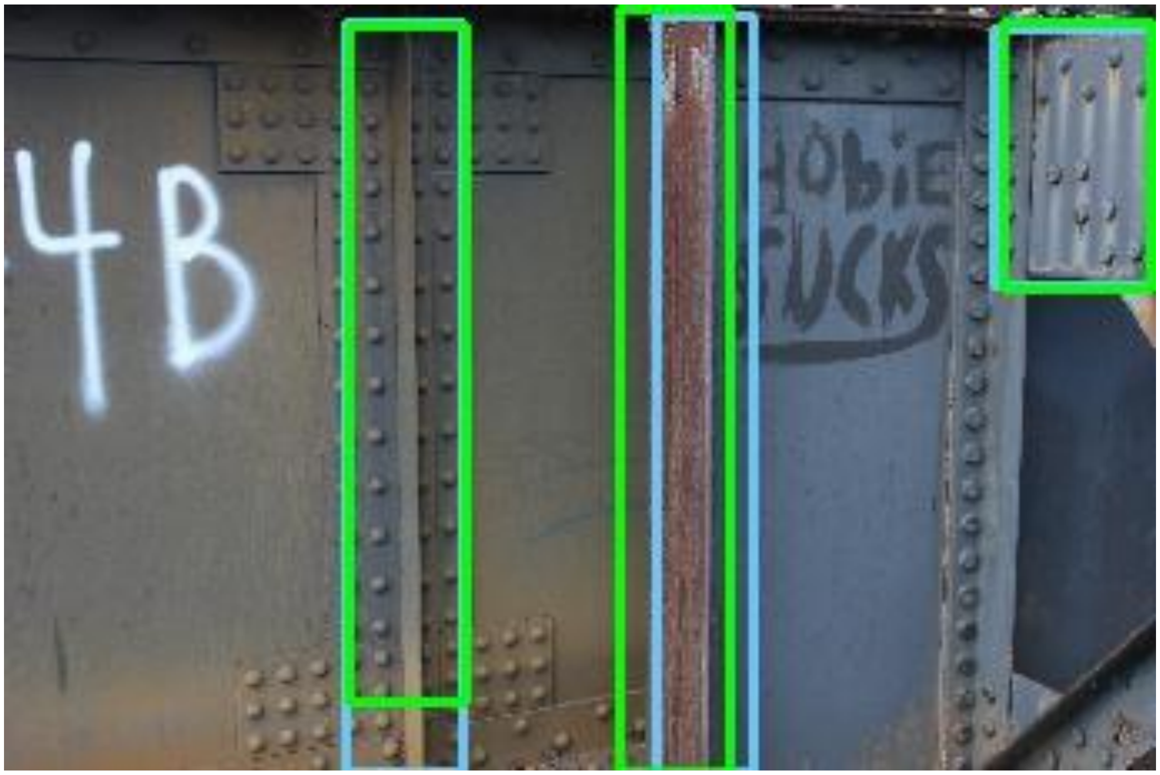


Figure D.2

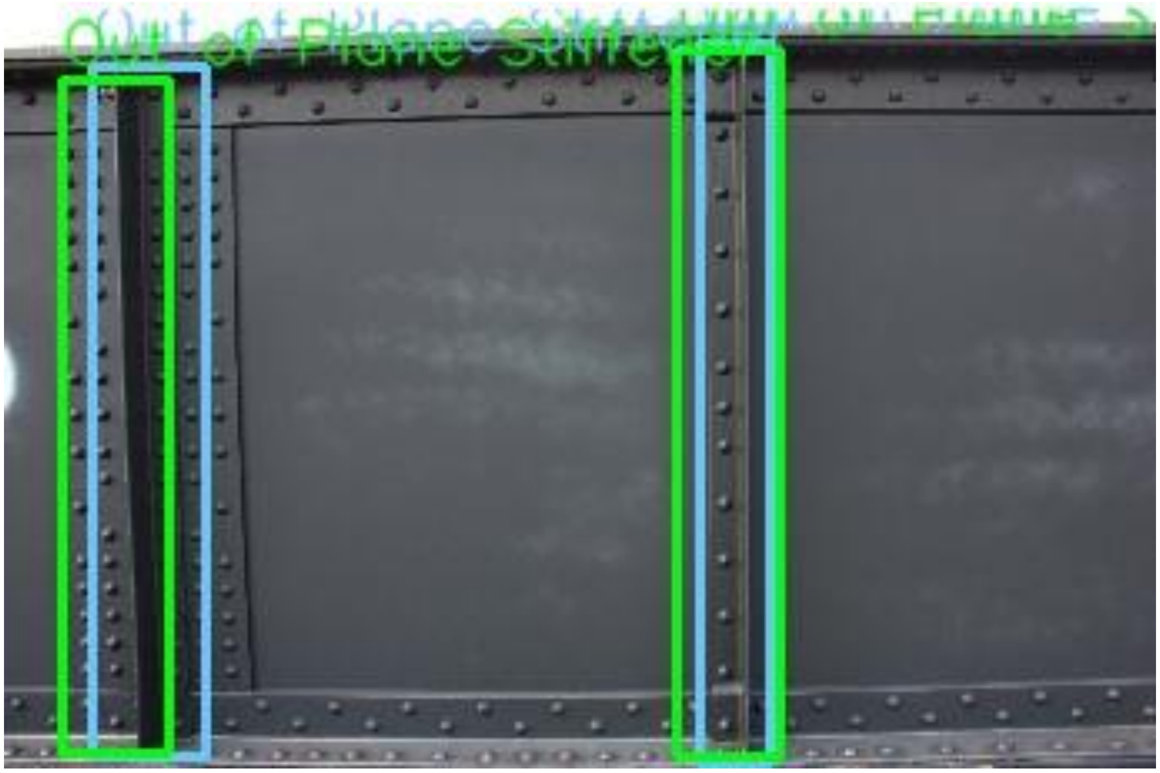


Figure D.3

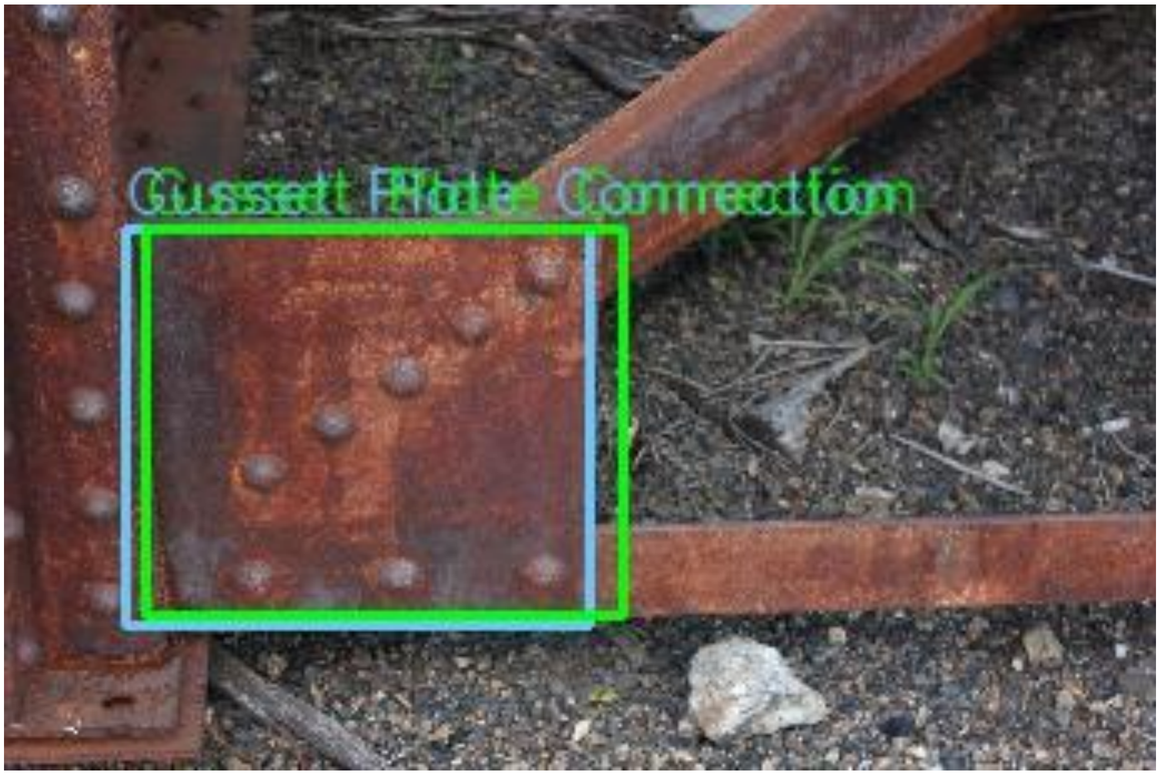


Figure D.4

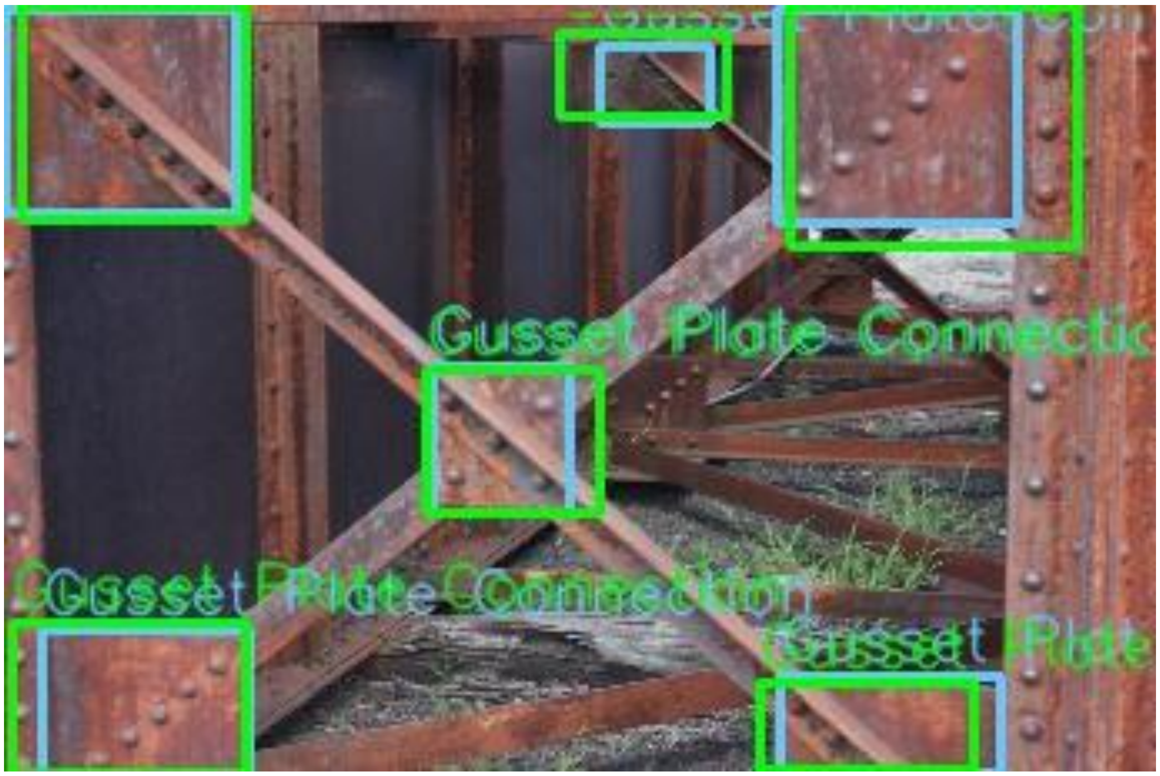


Figure D.5

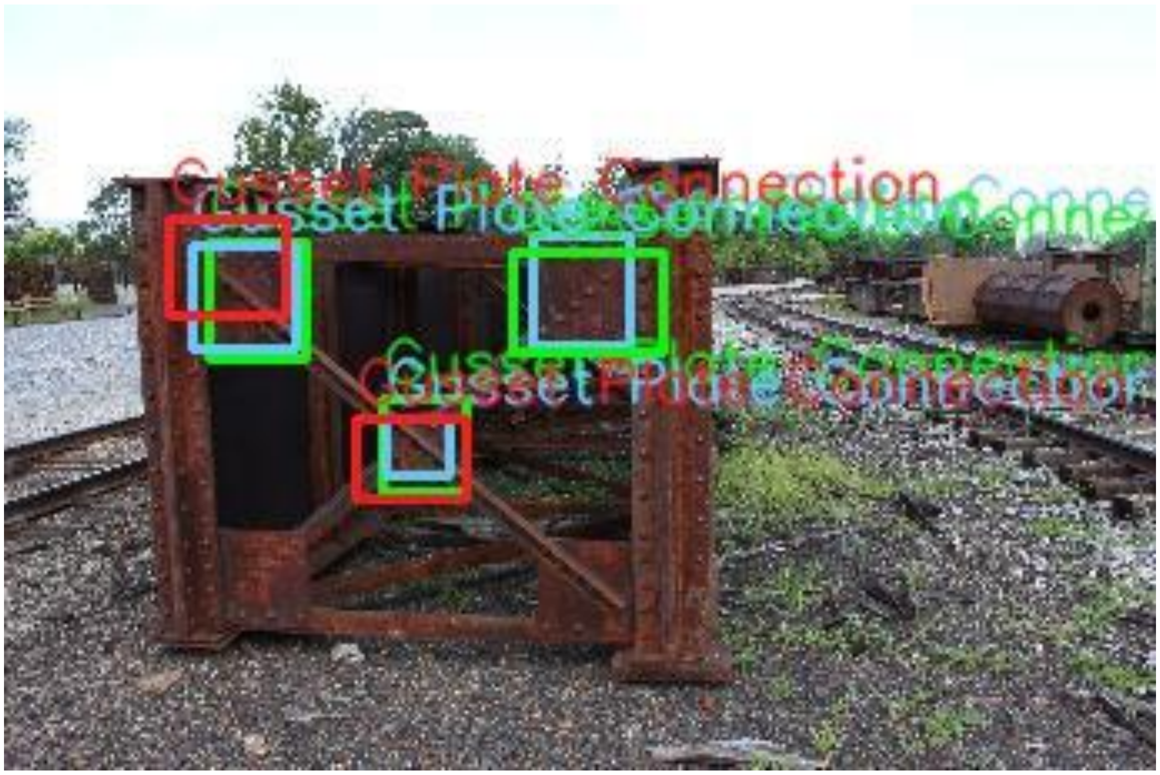

Gusset Plate Connection
Gusset Plate Connection
Gusset Plate Connection

Figure D.6

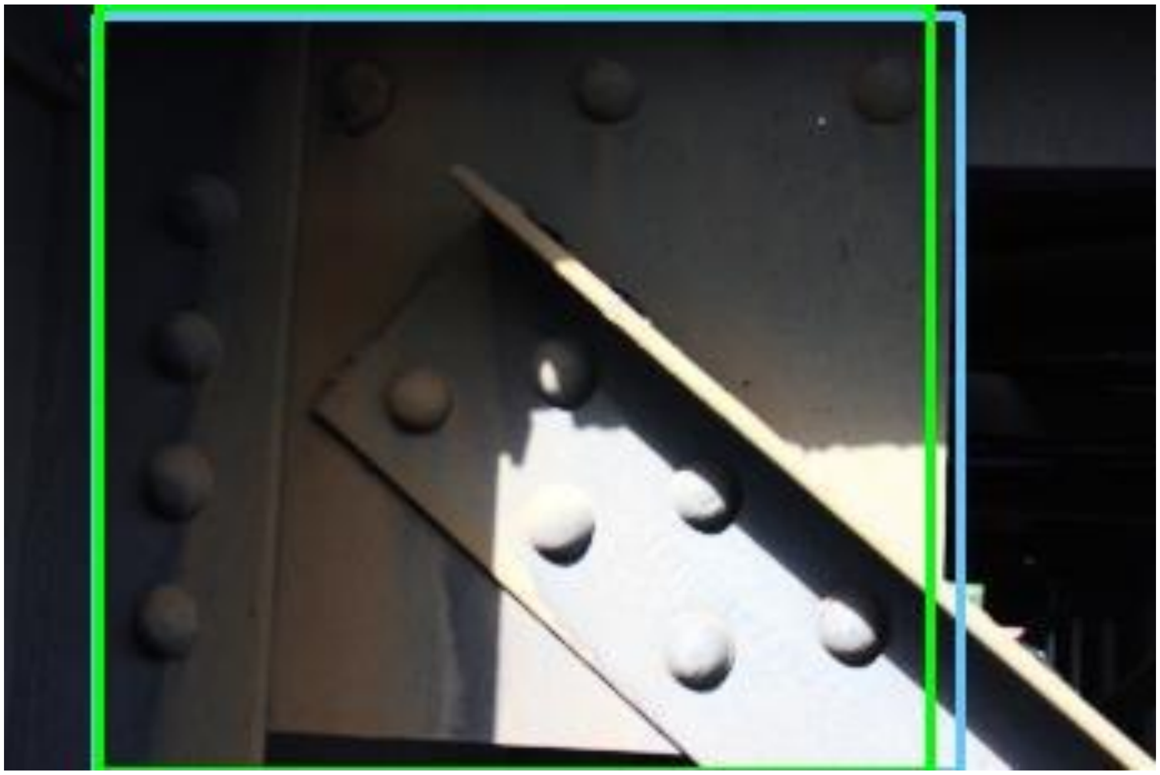


Figure D.7

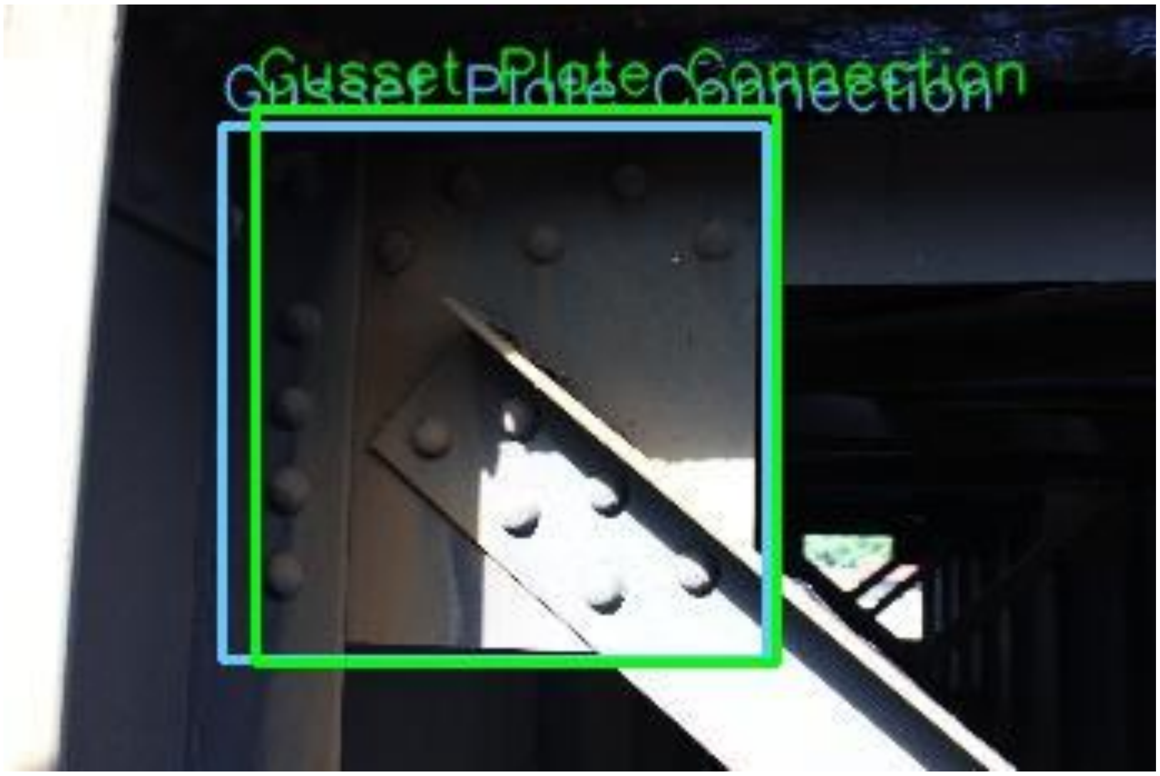


Figure D.8

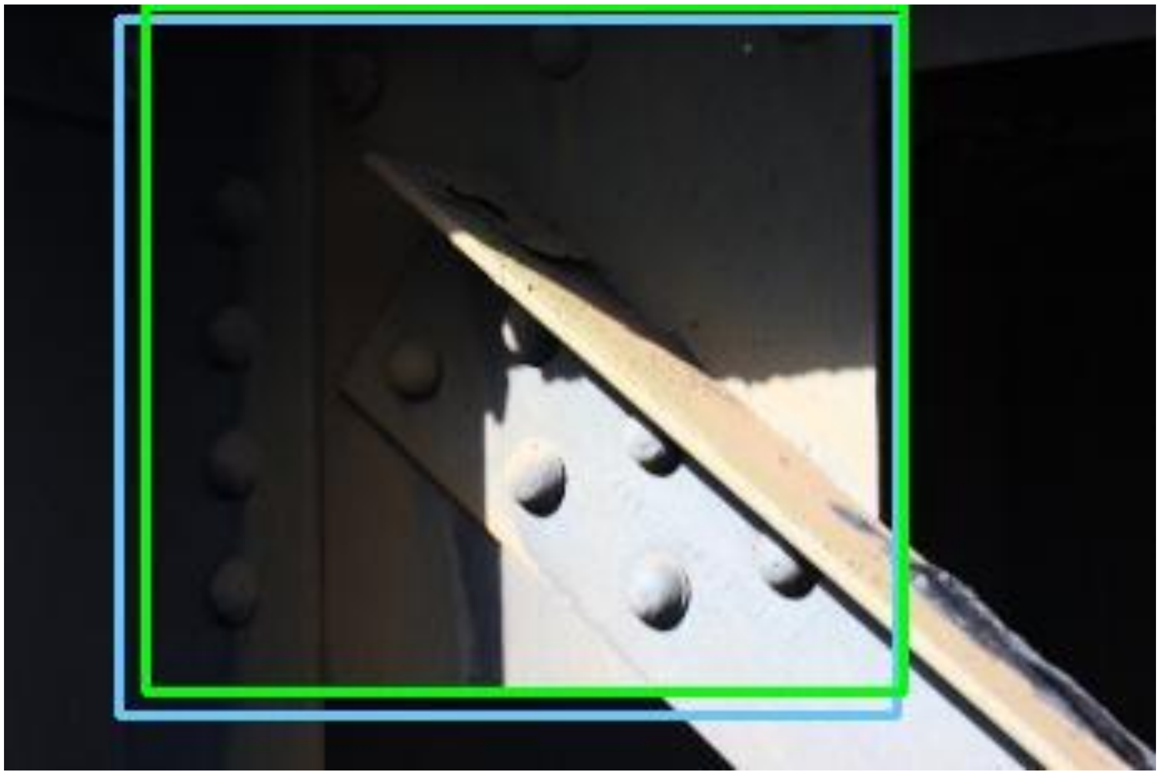


Figure D.9

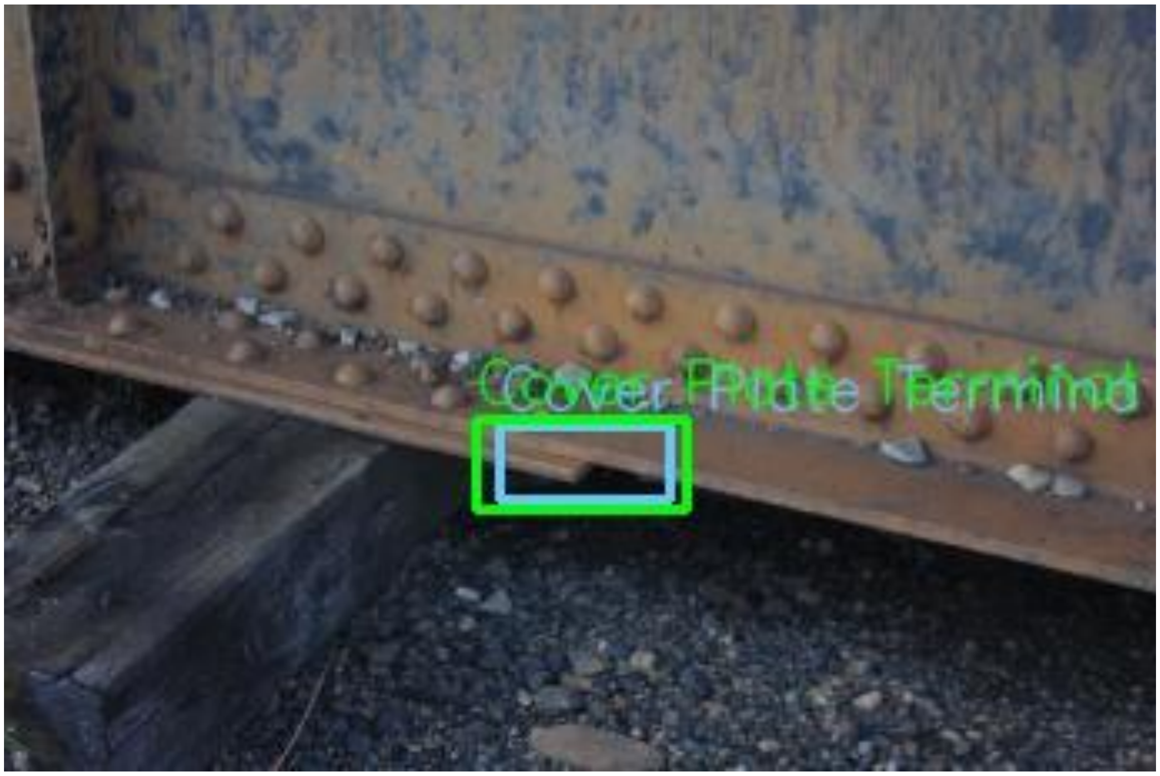


Figure D.10

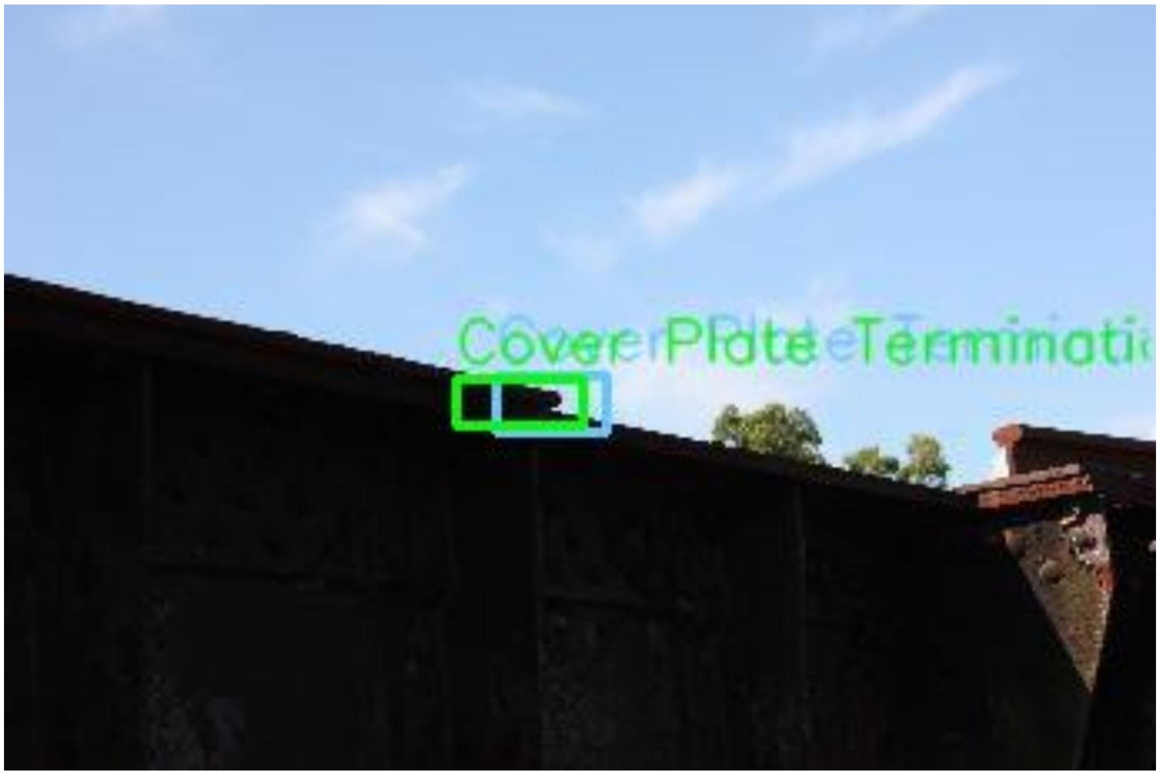


Figure D.11

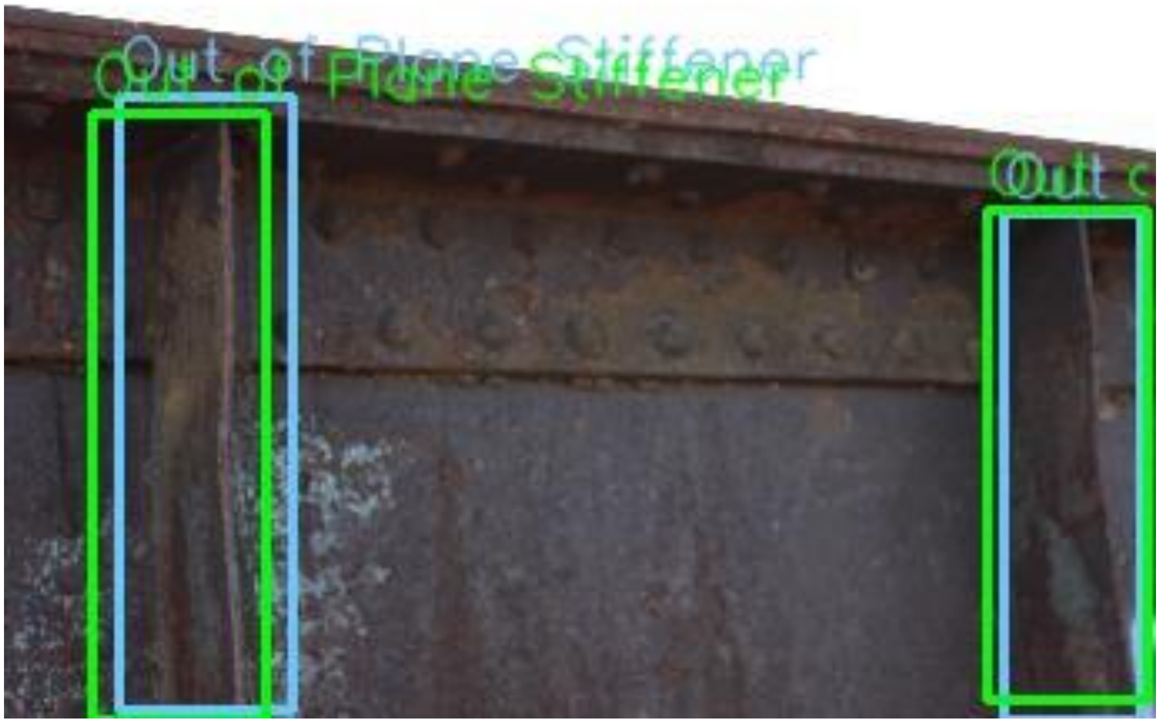


Figure D.12

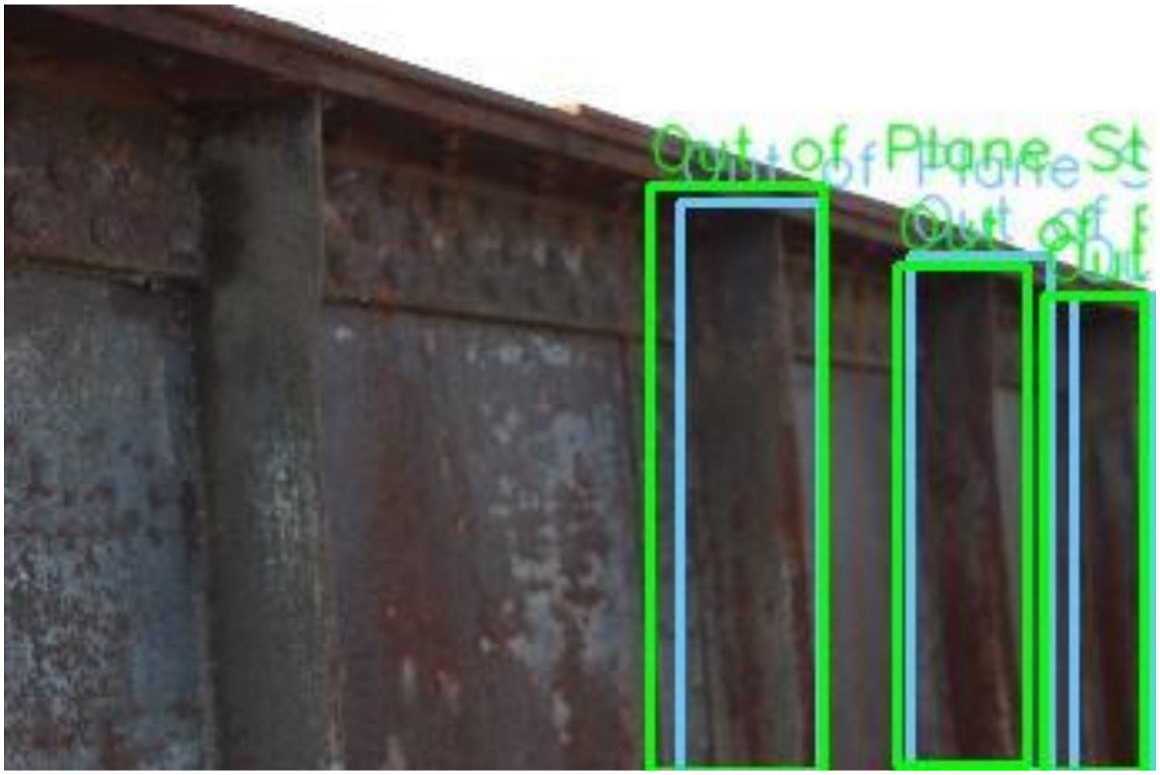


Figure D.13

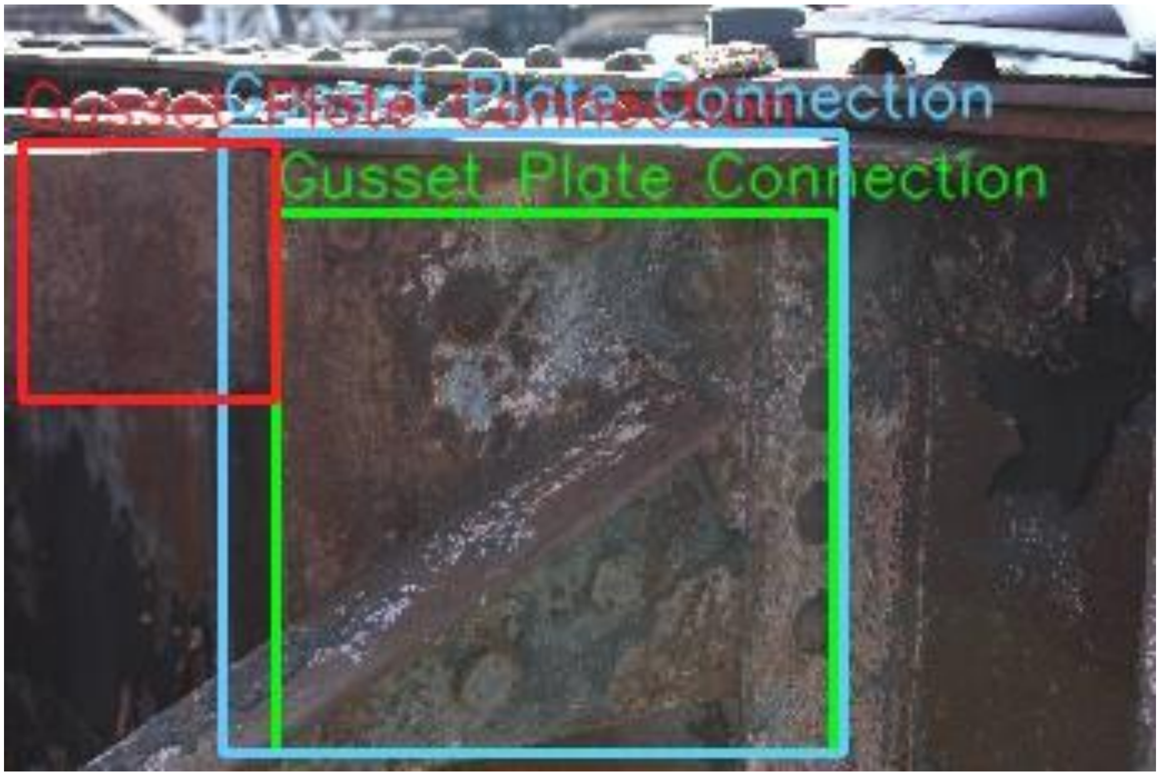


Figure D.14

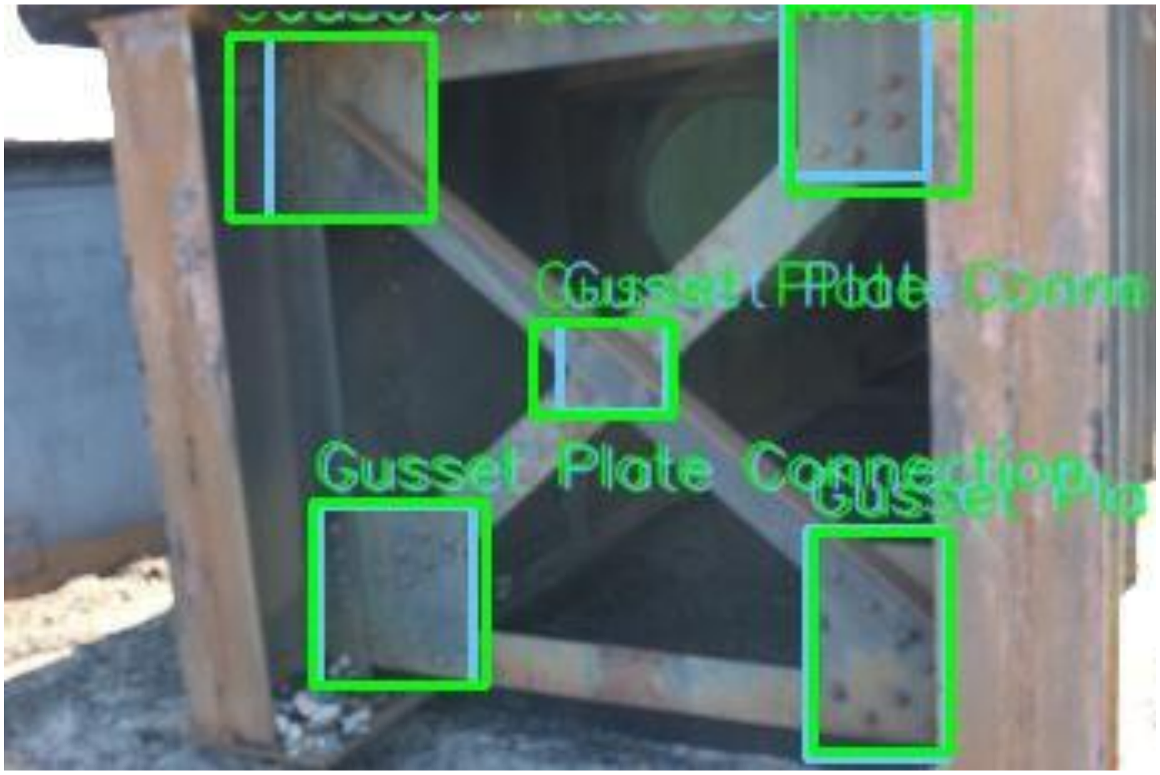


Figure D.15

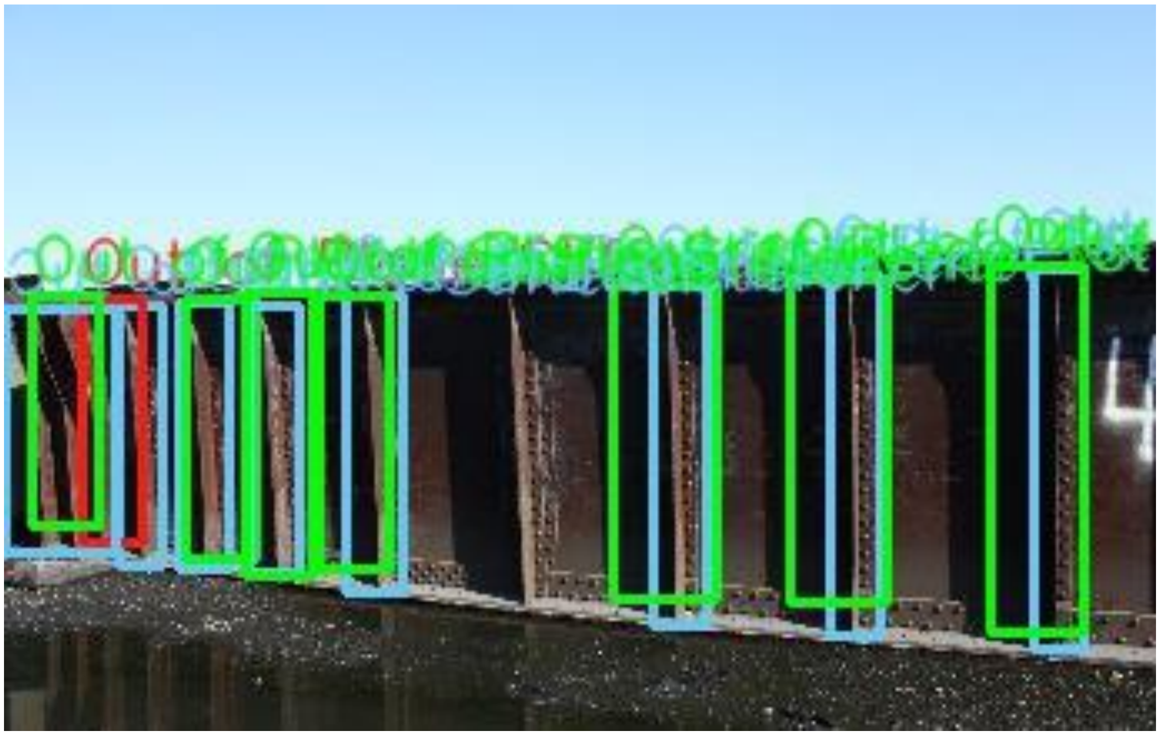


Figure D.16

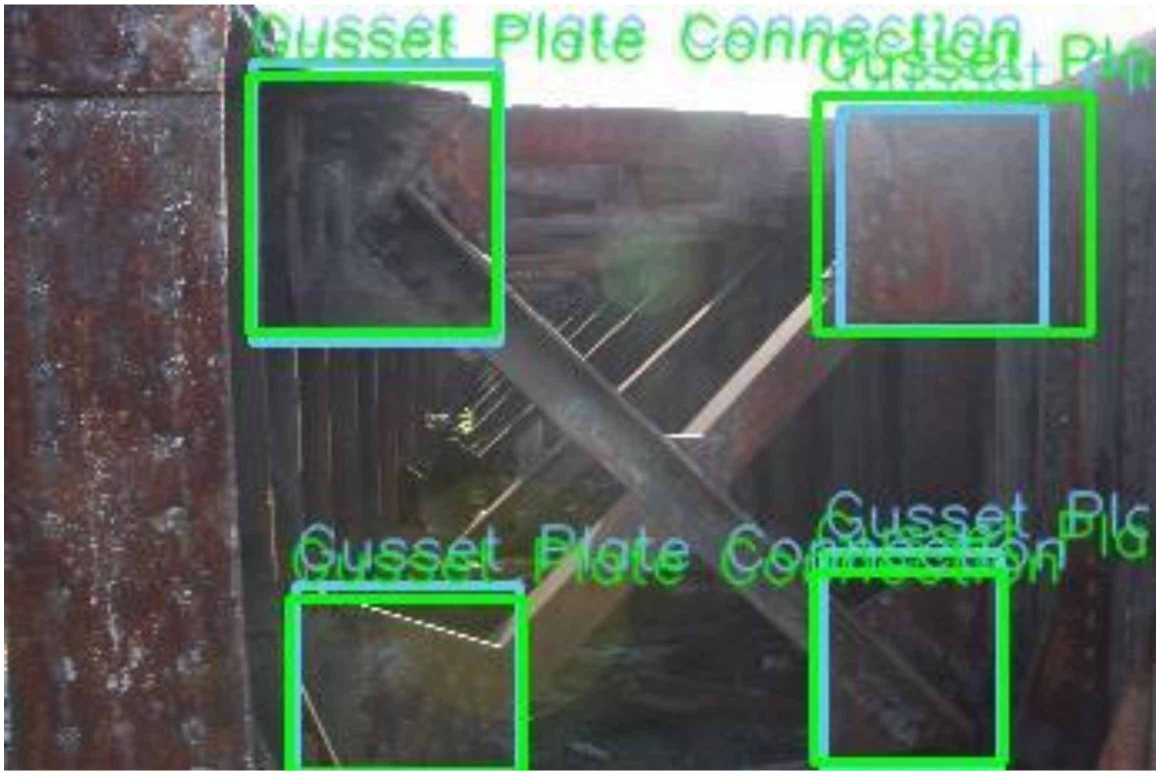


Figure D.17

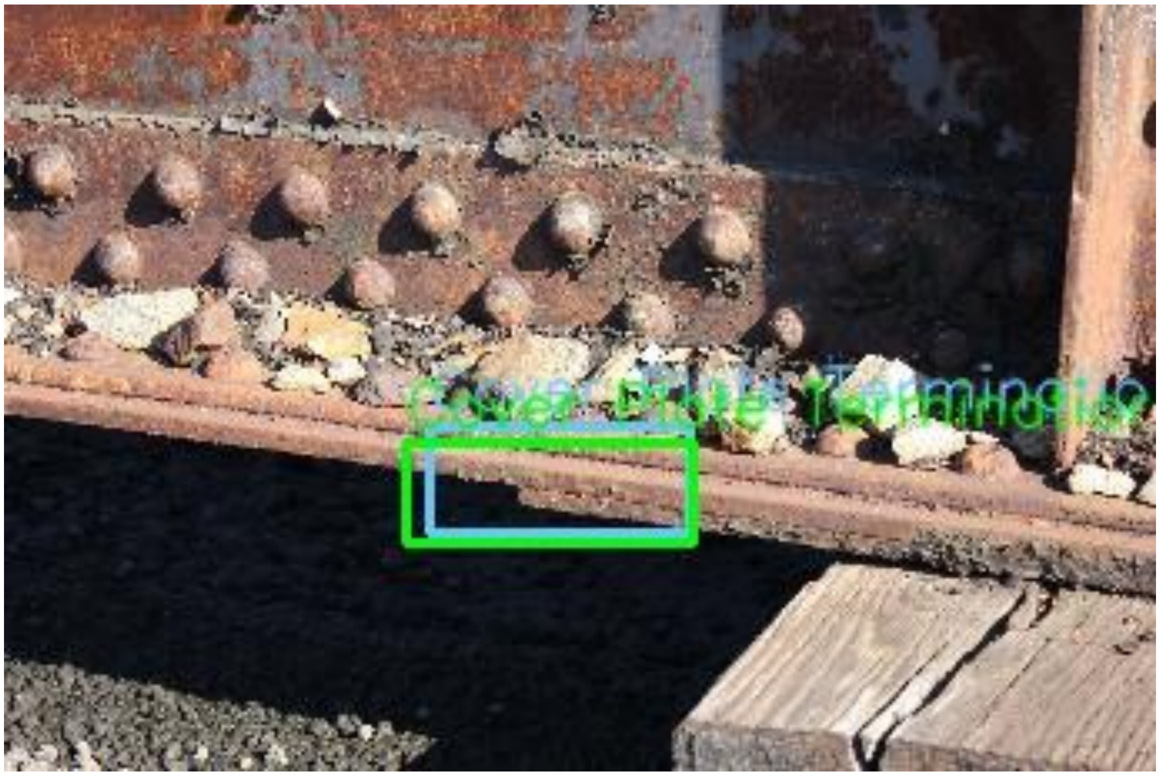


Figure D.18

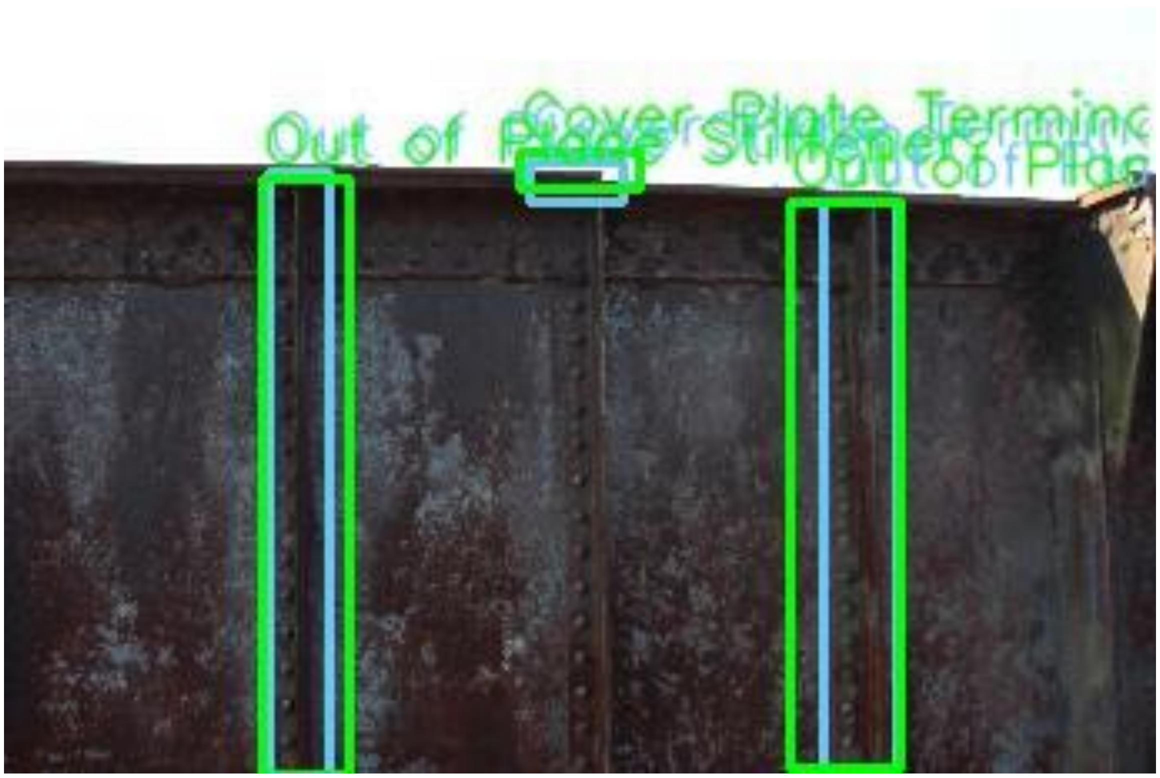


Figure D.19

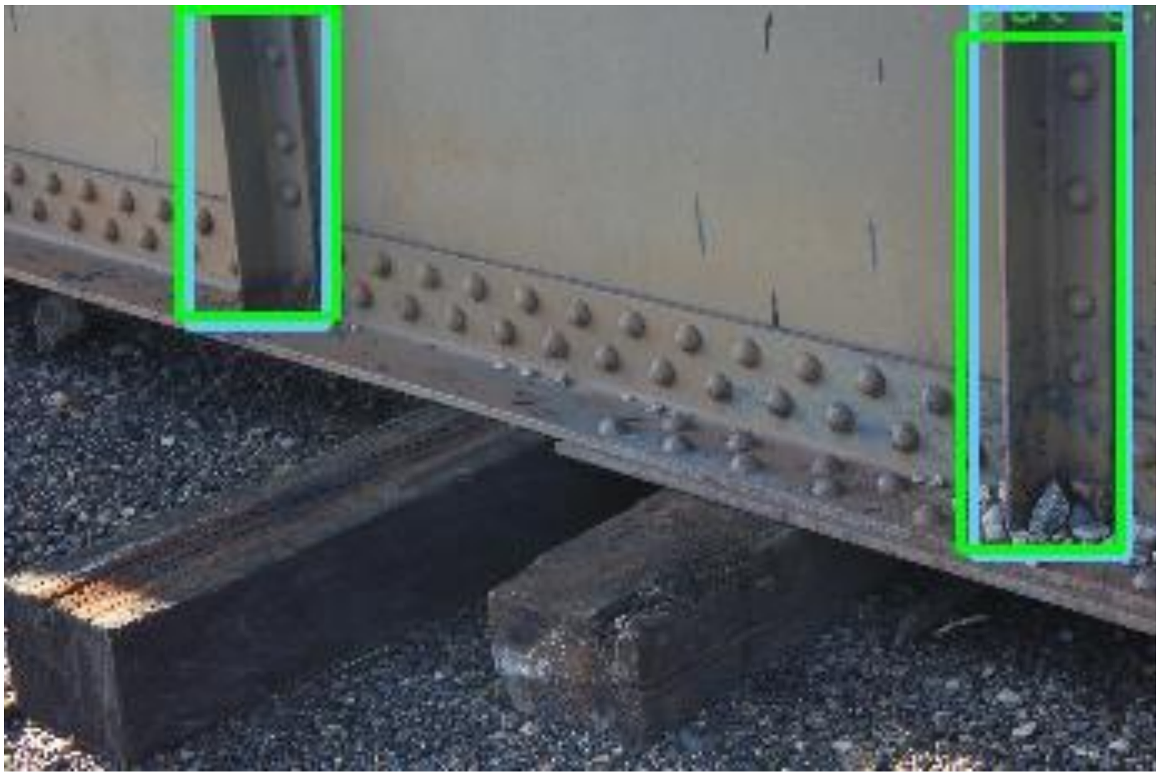


Figure D.20

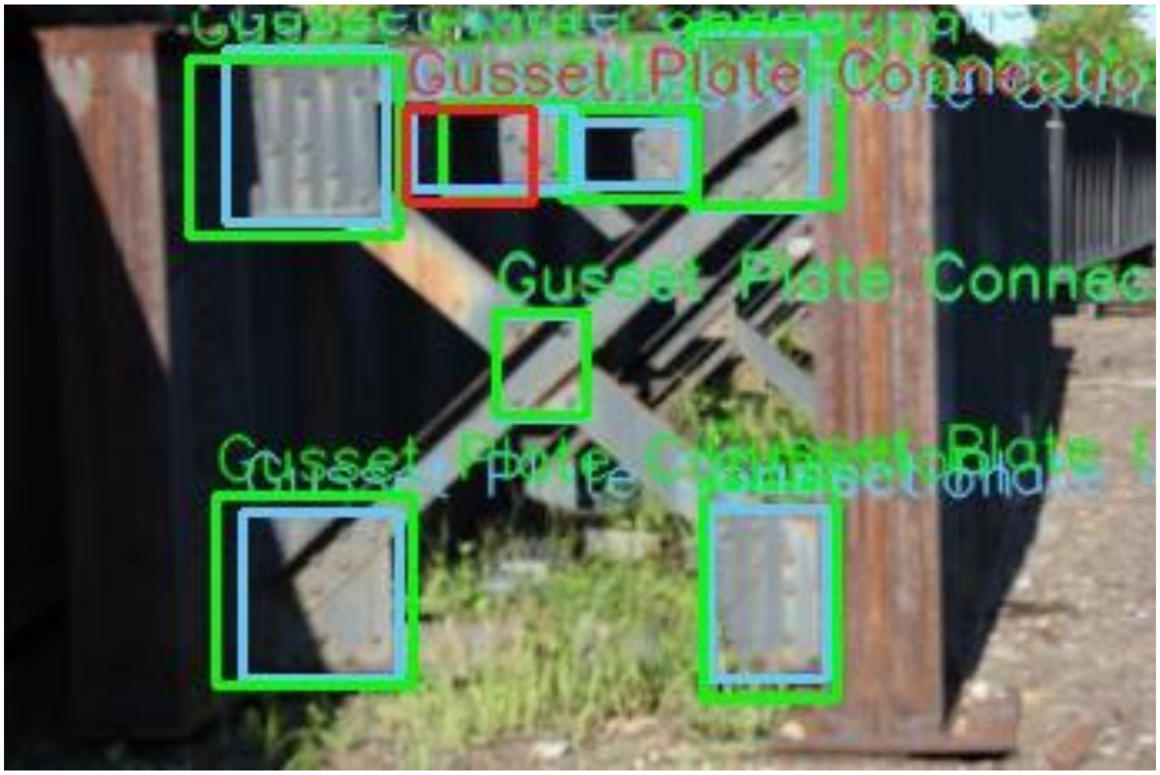


Figure D.21

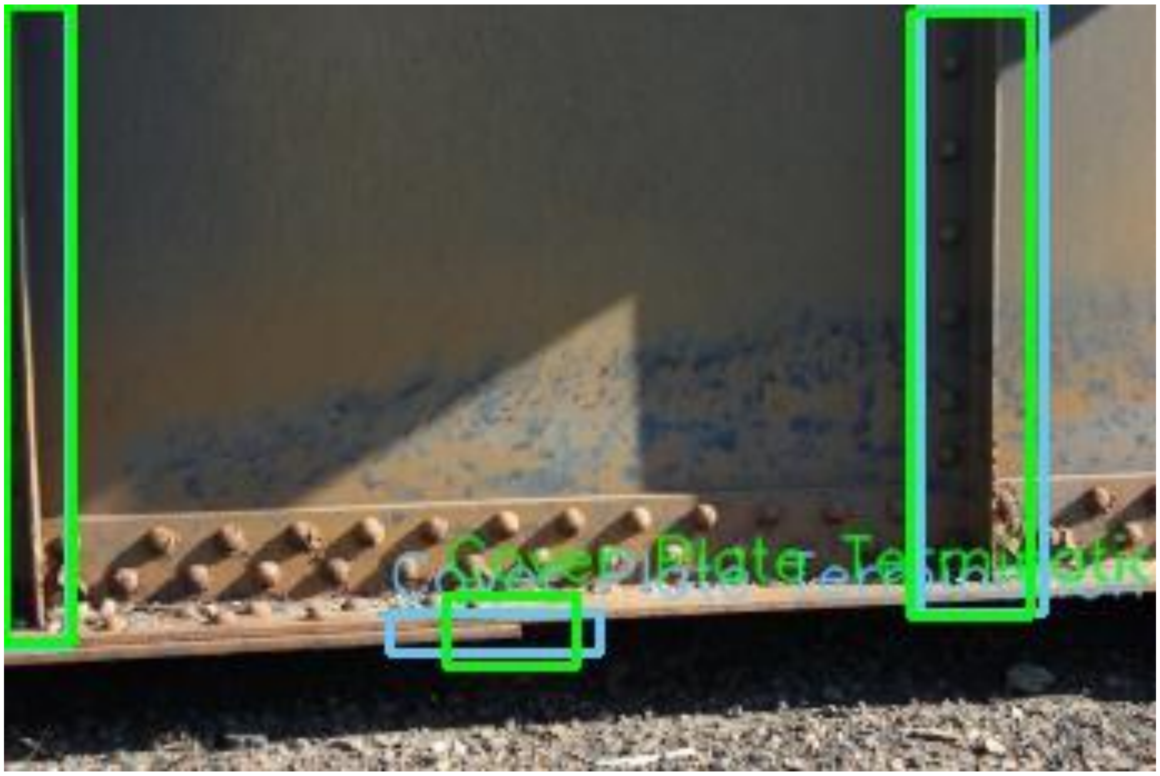


Figure D.22

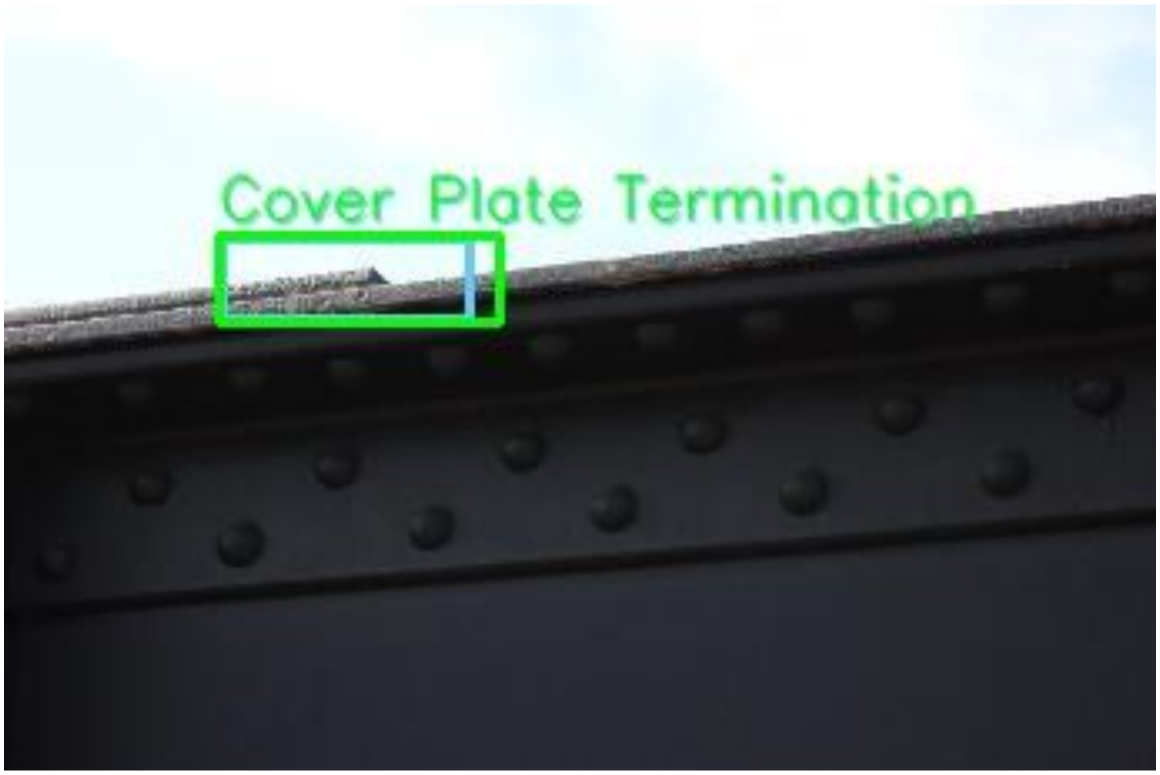


Figure D.23

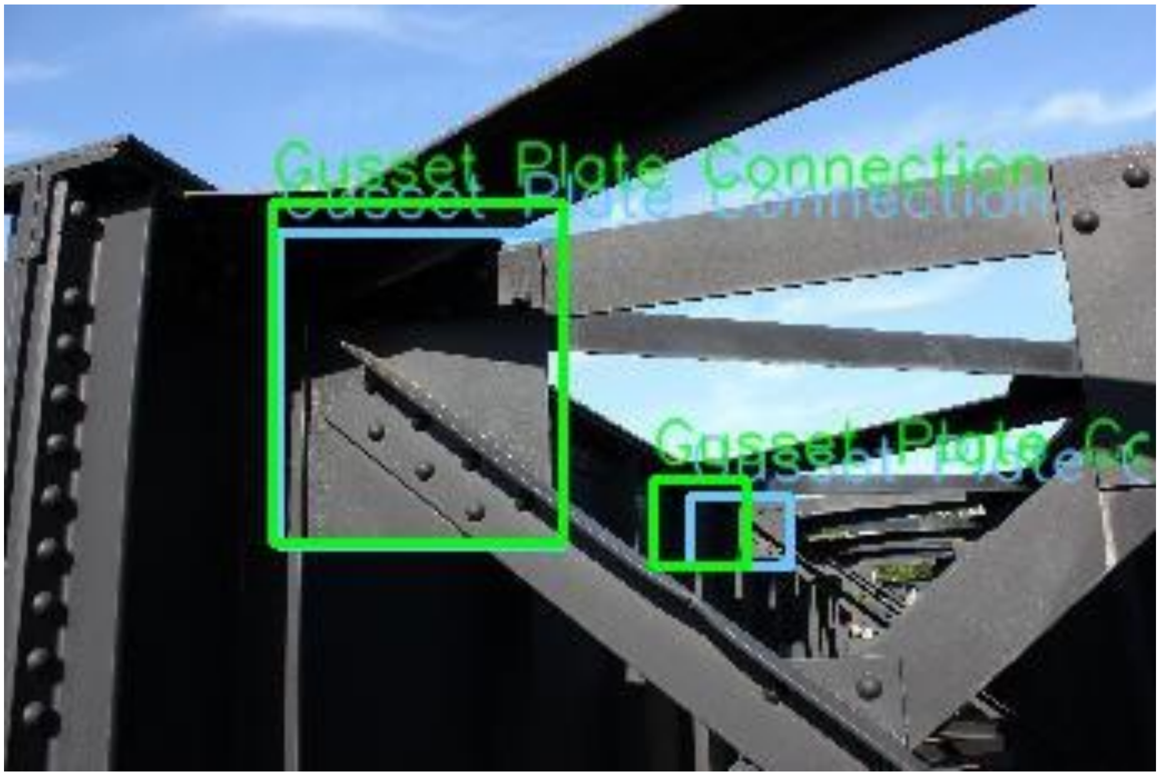


Figure D.24

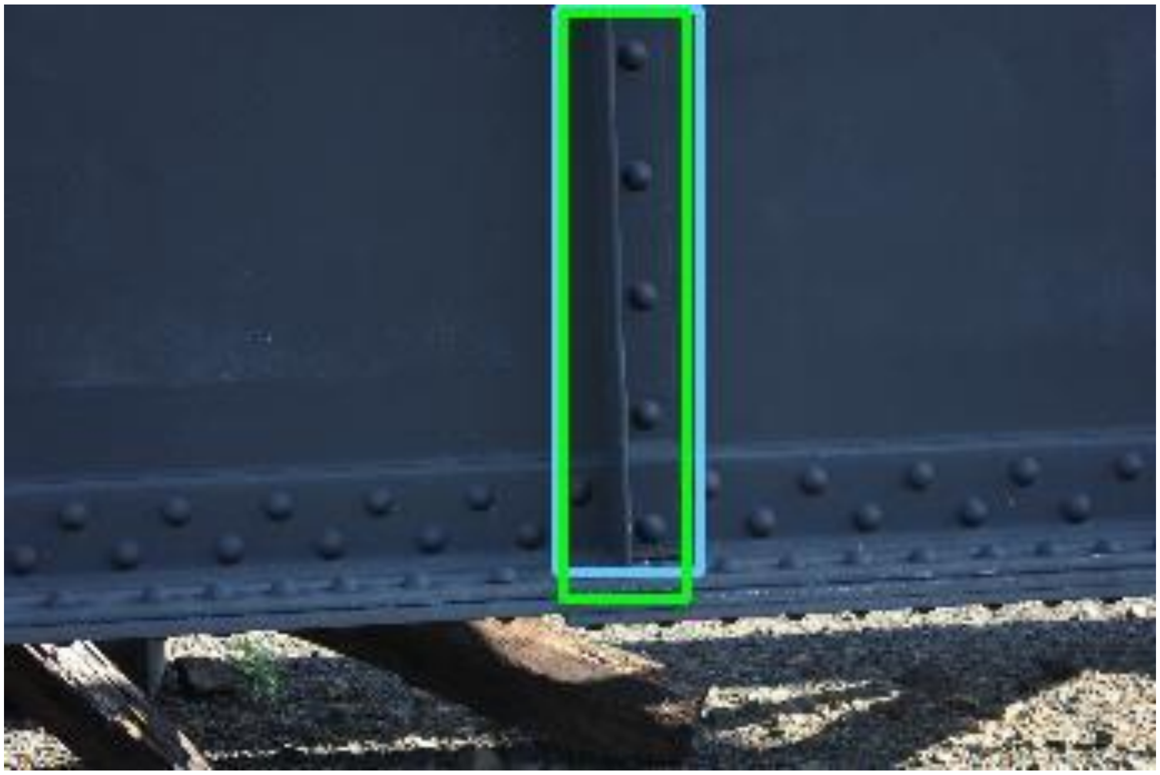


Figure D.25

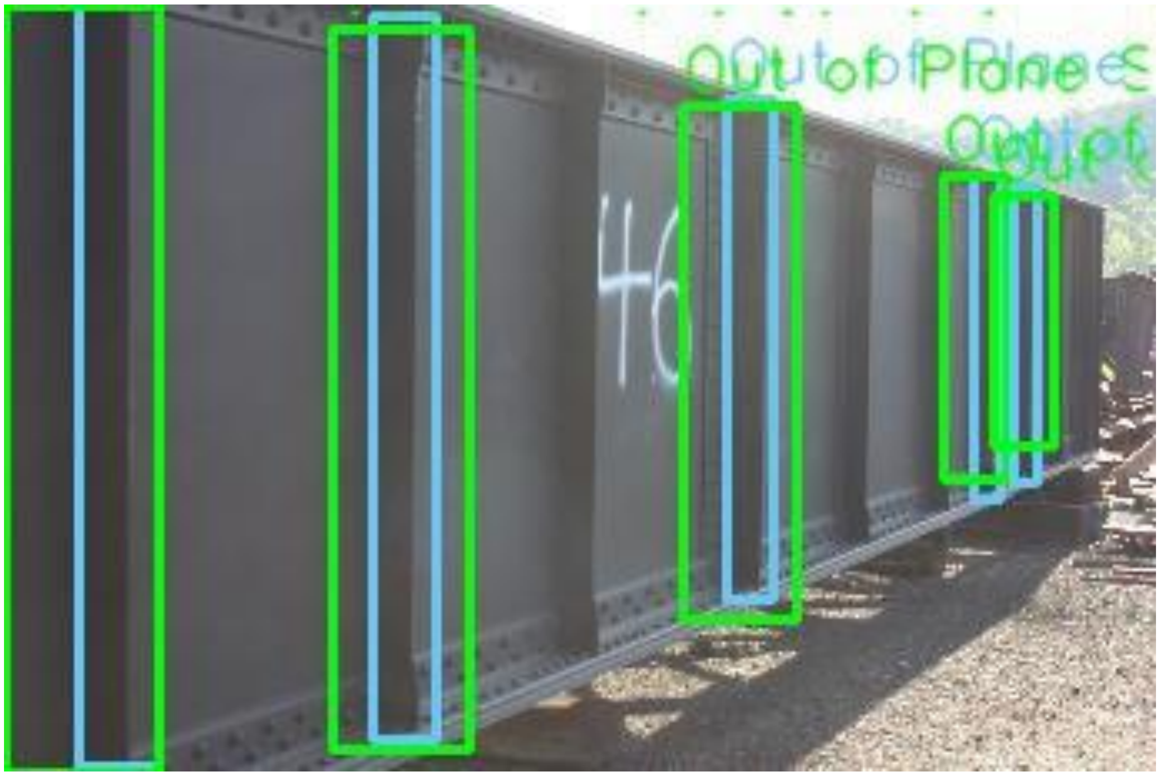


Figure D.26

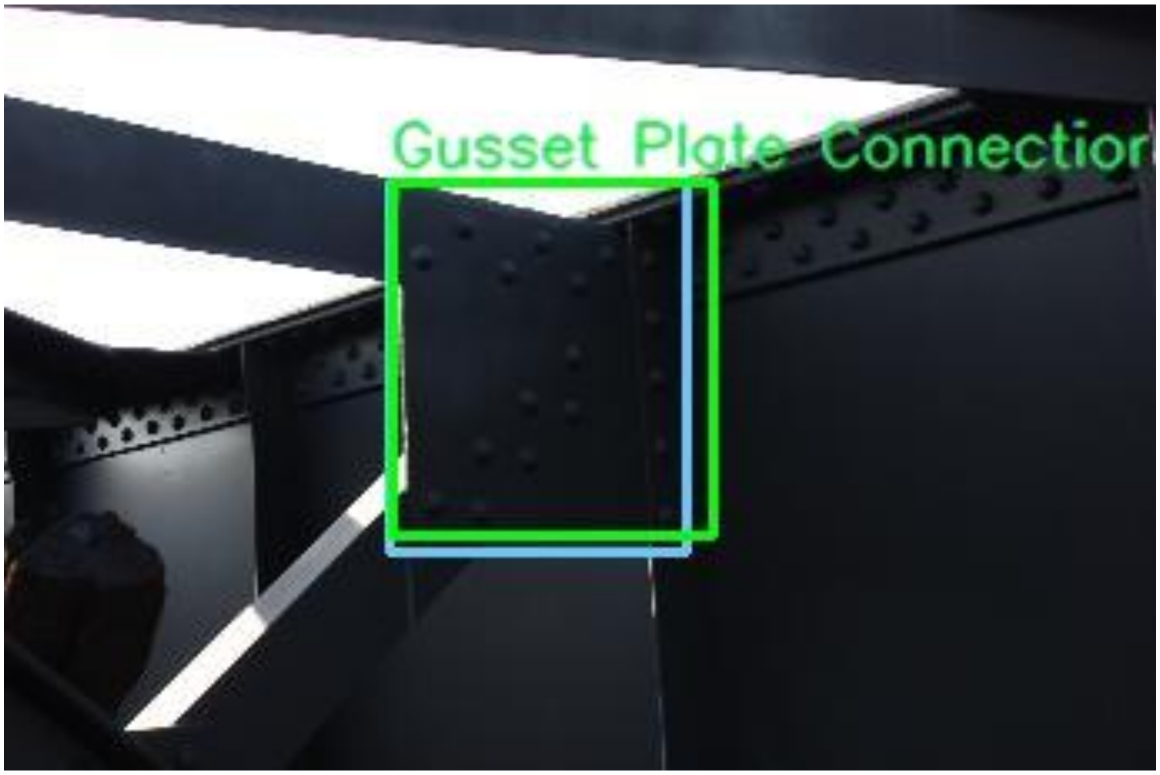


Figure D.27

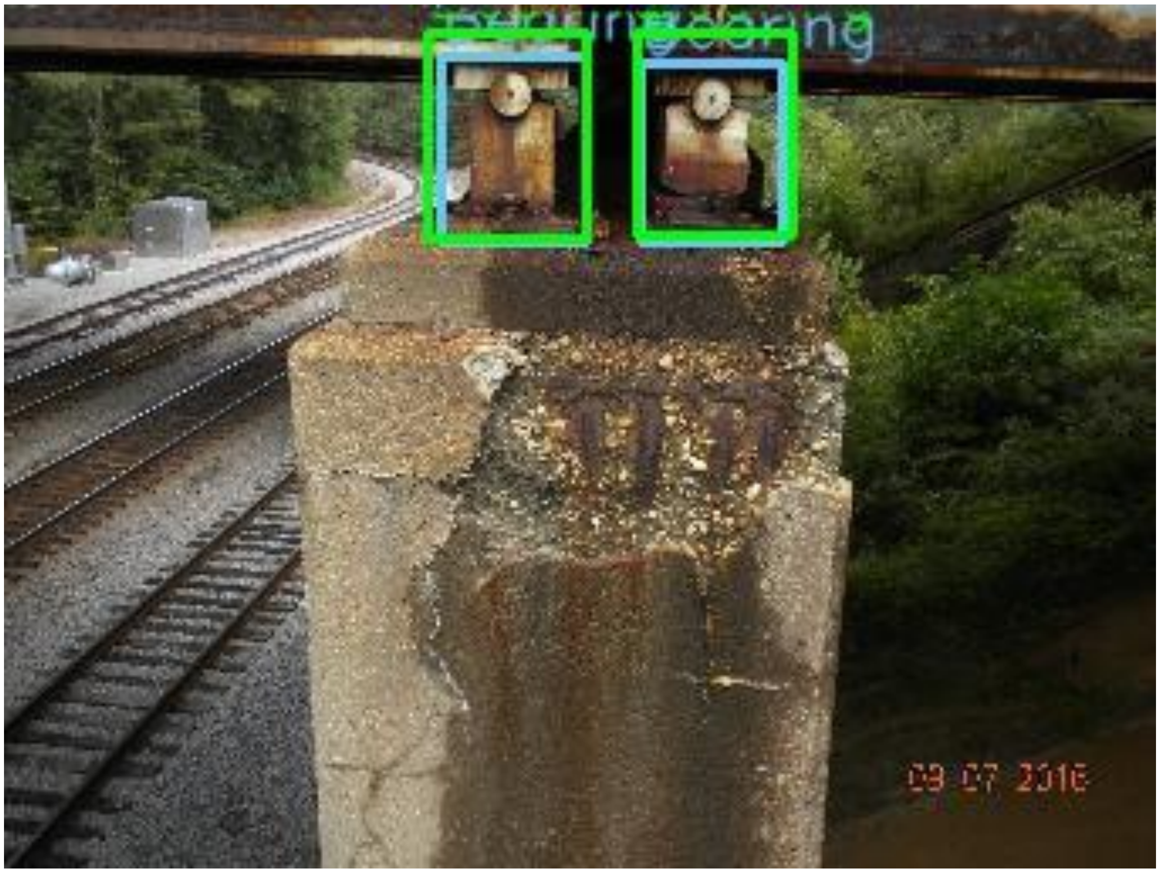


Figure D.28

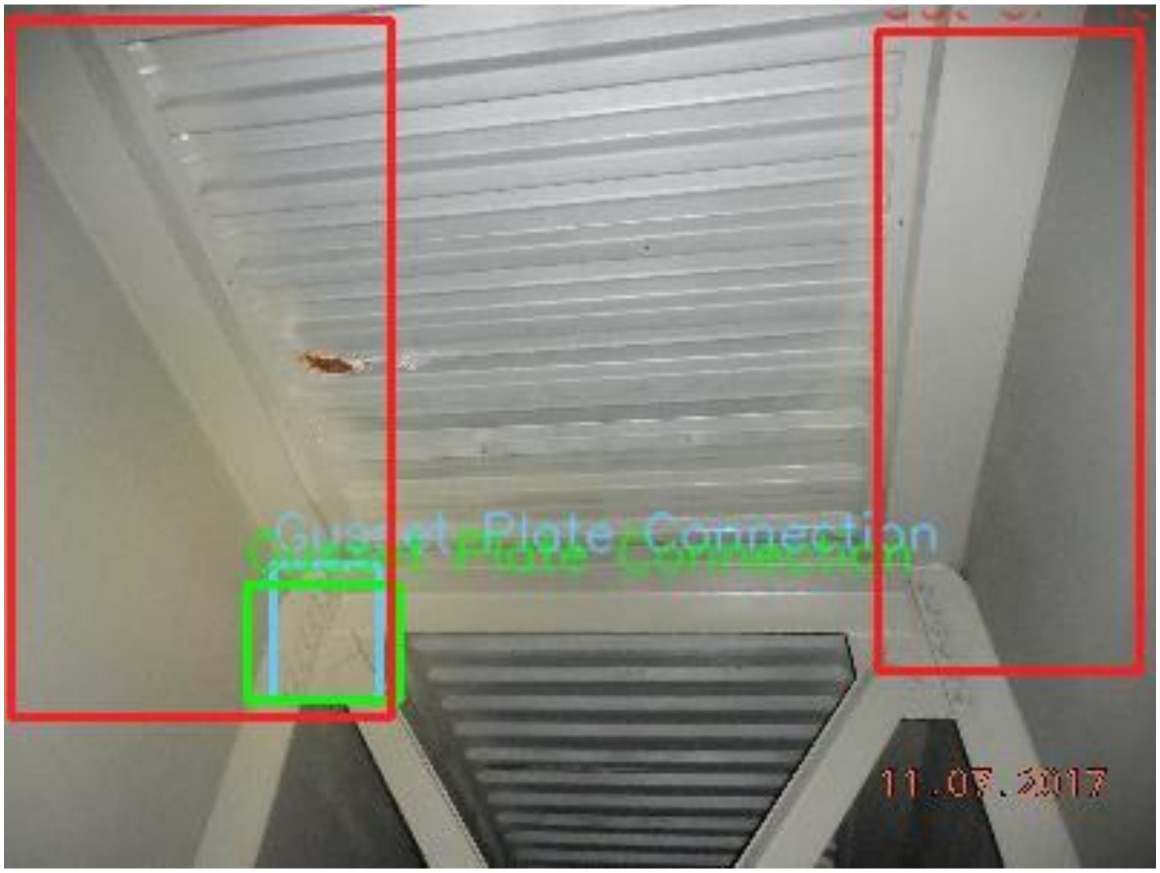

Gusset Plate Connection
Gusset Plate Connection
11.07.2017

Figure D.29

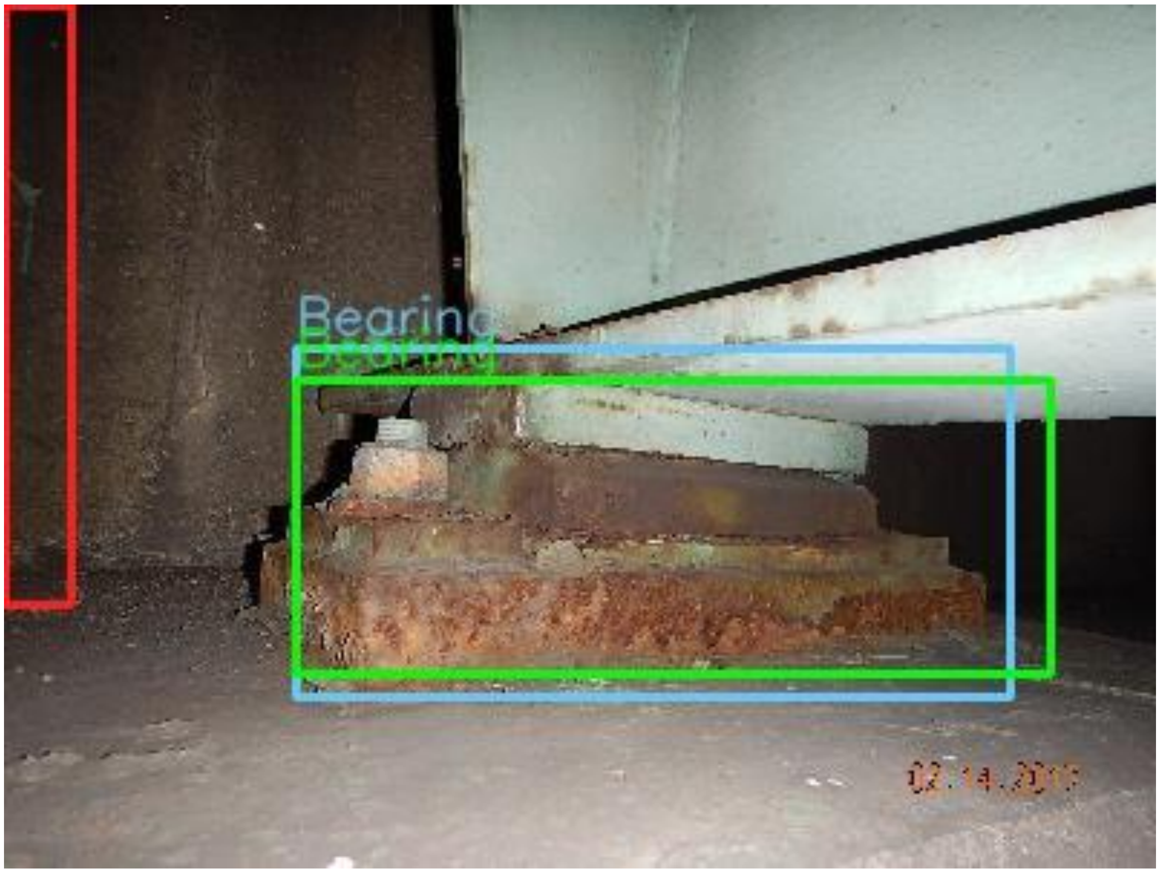


Figure D.30

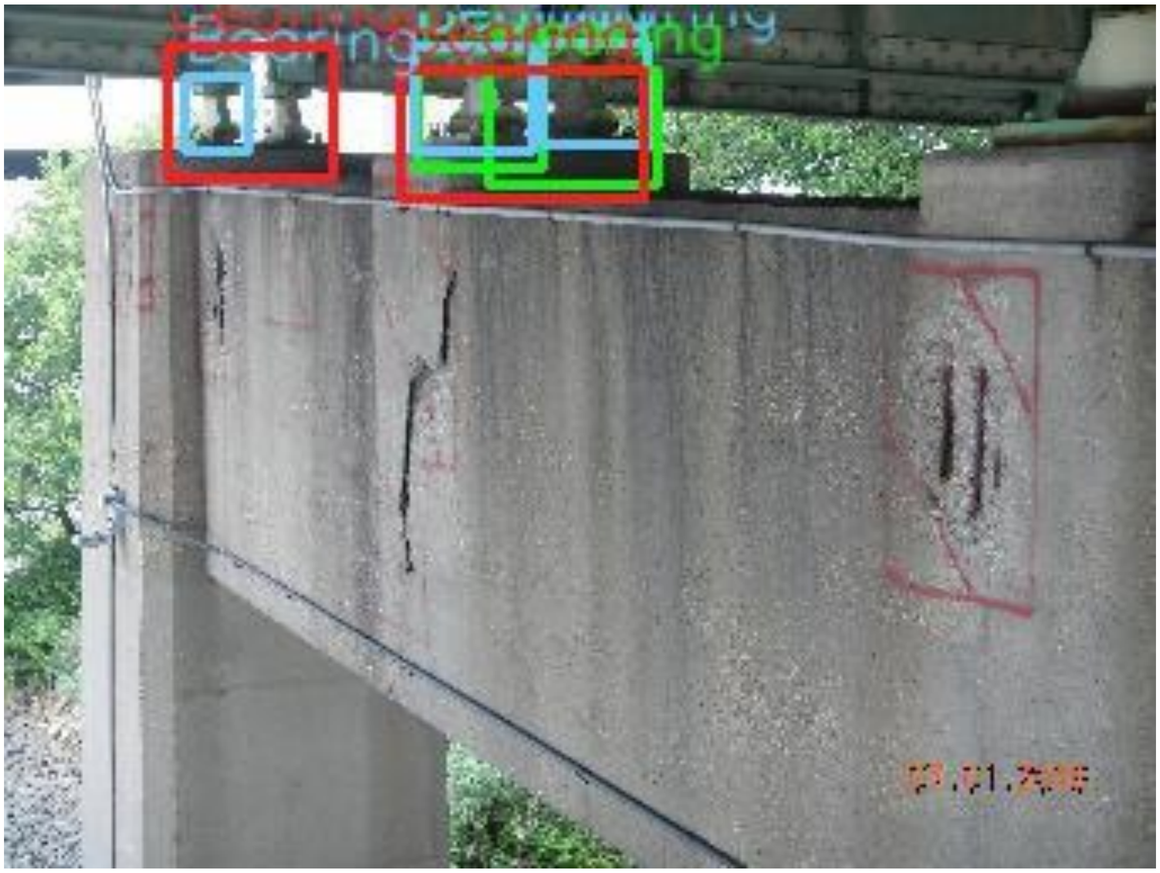


Figure D.31

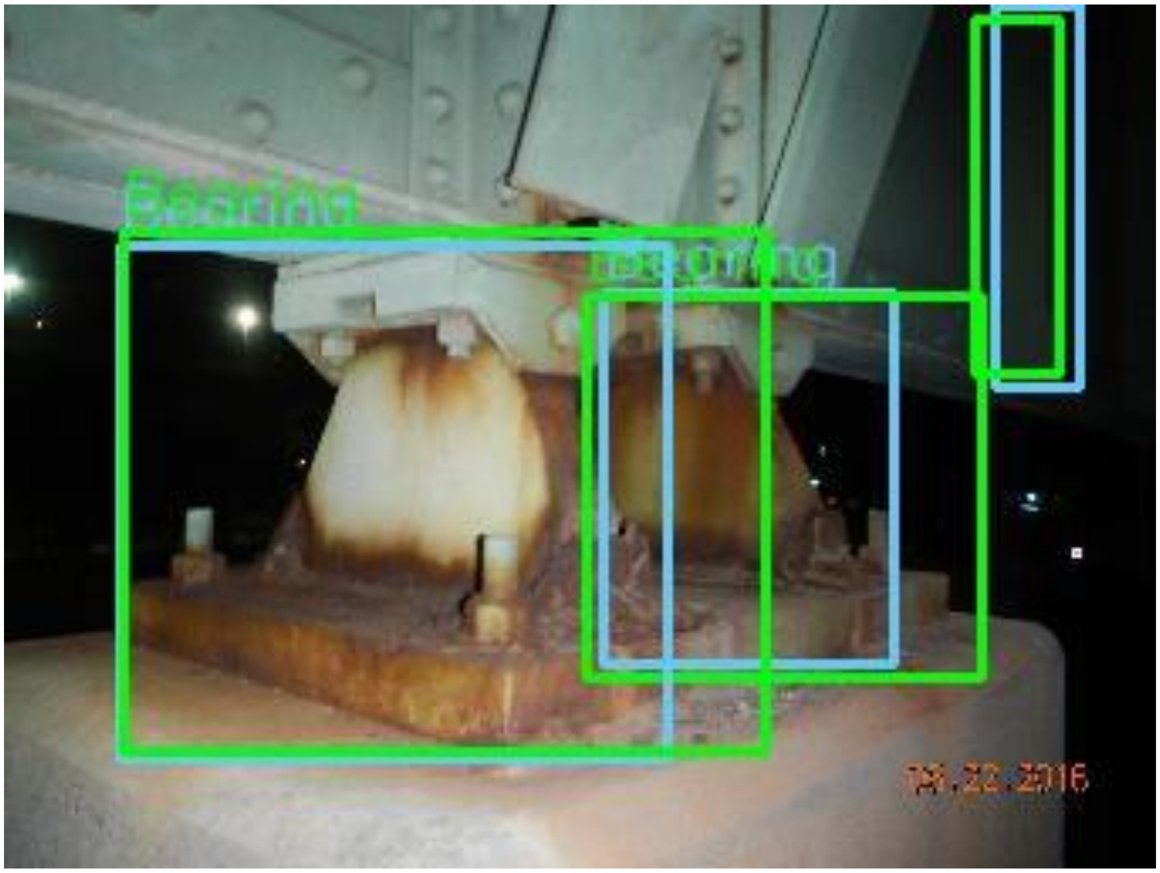


Figure D.32

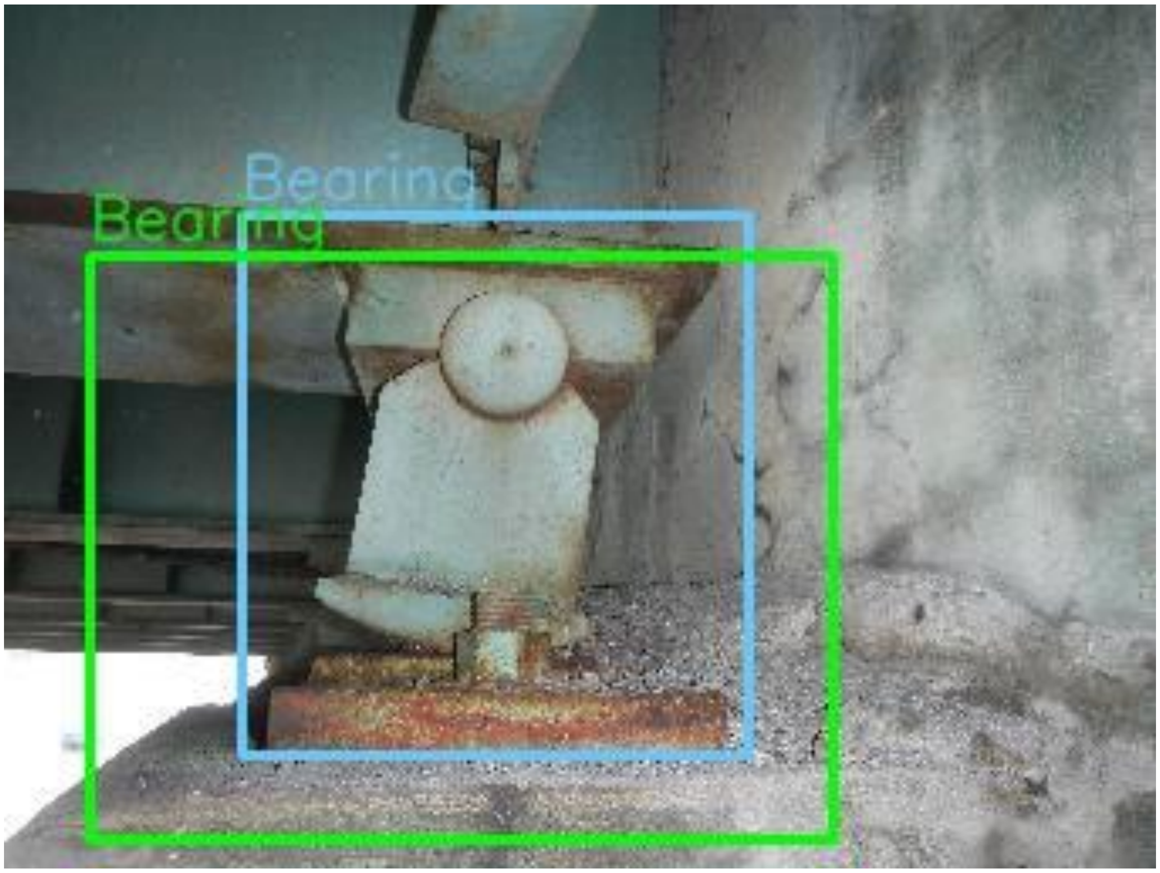


Figure D.33

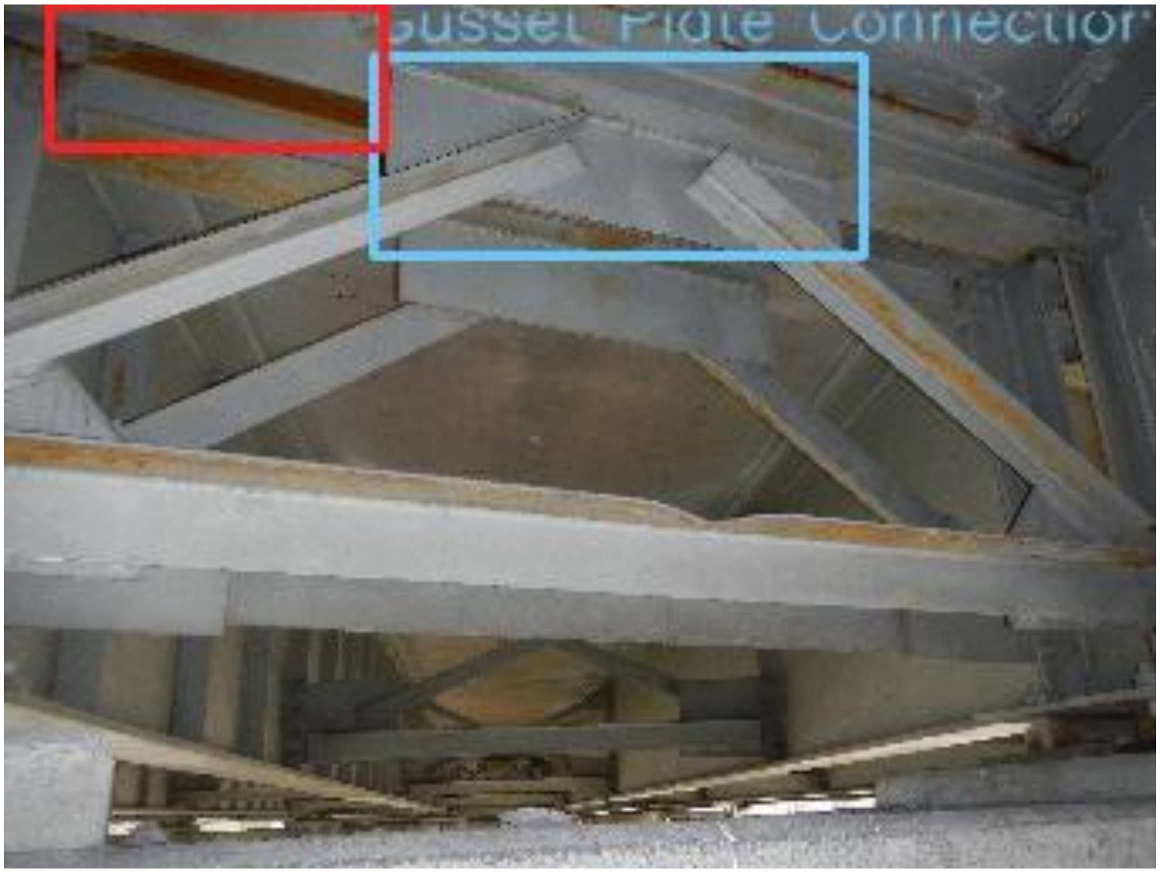


Figure D.34

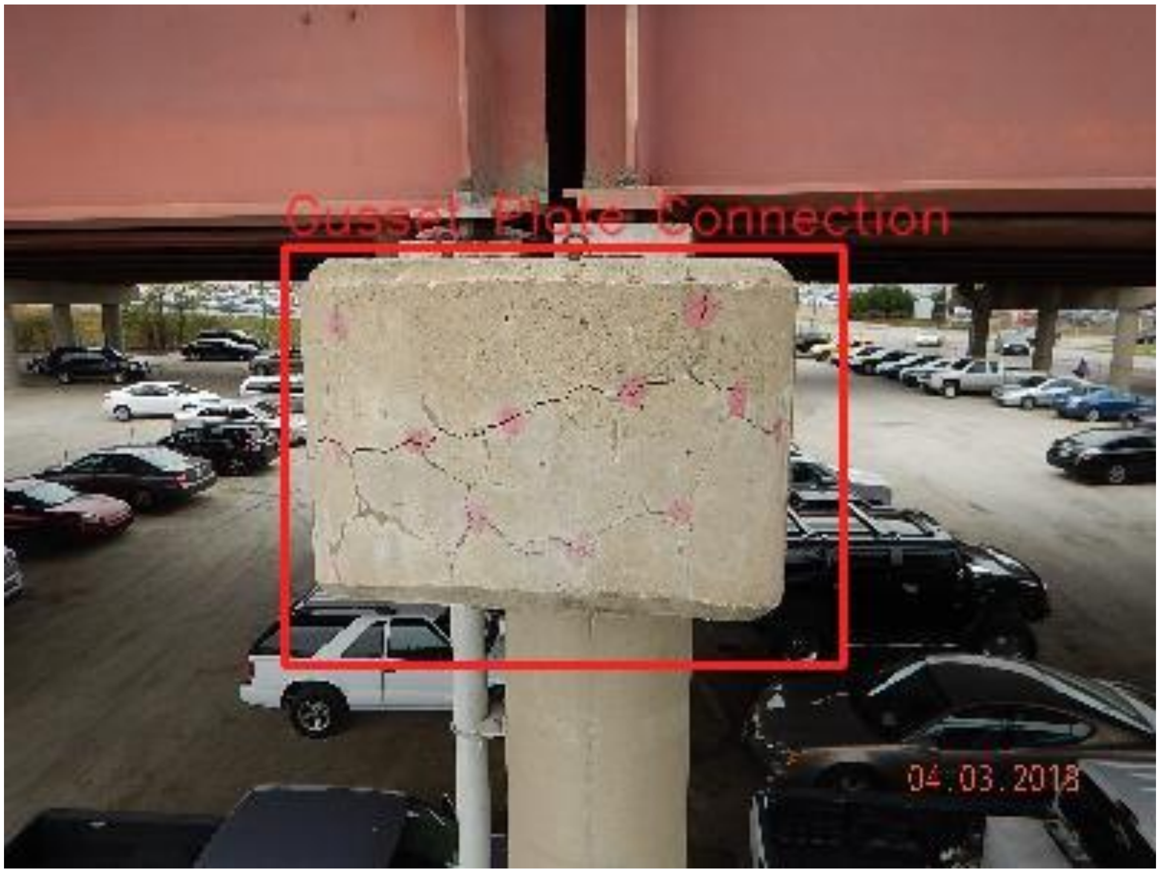


Figure D.35

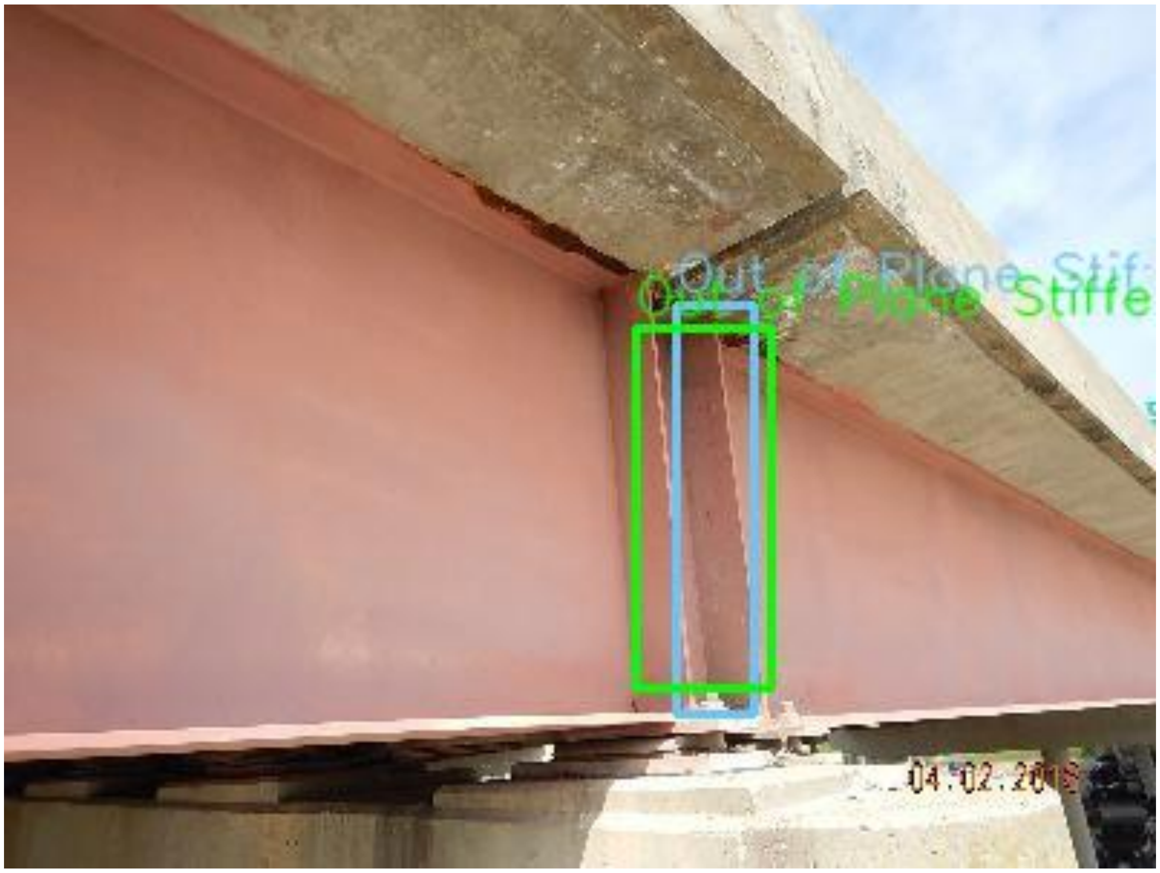


Figure D.36

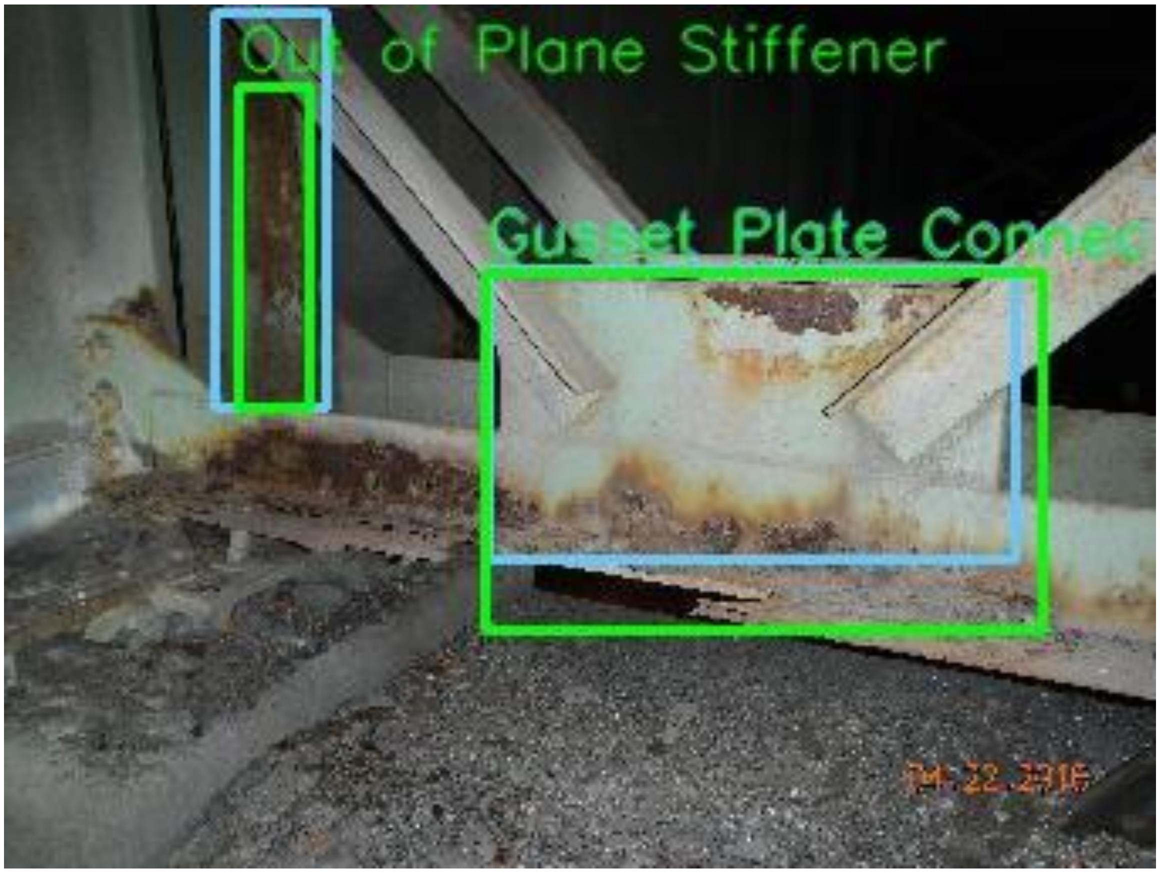


Figure D.37

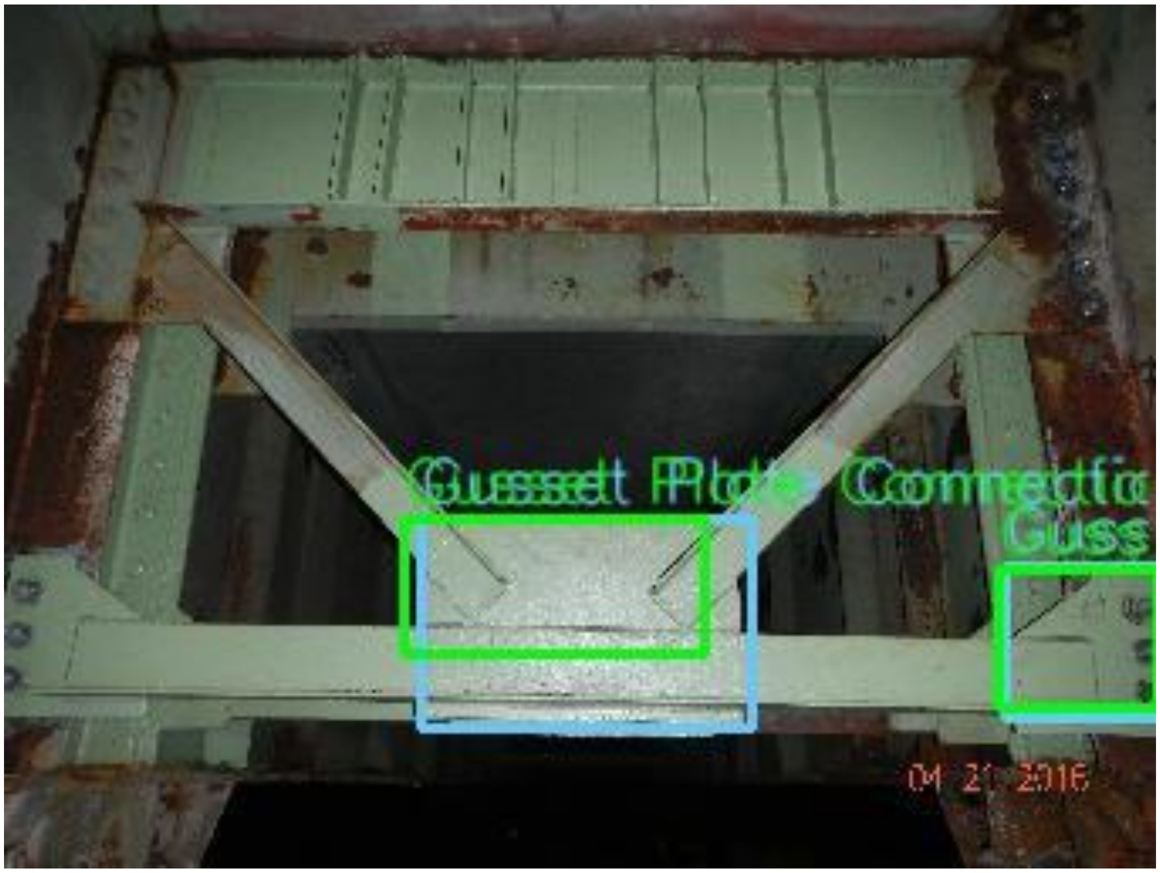


Figure D.38

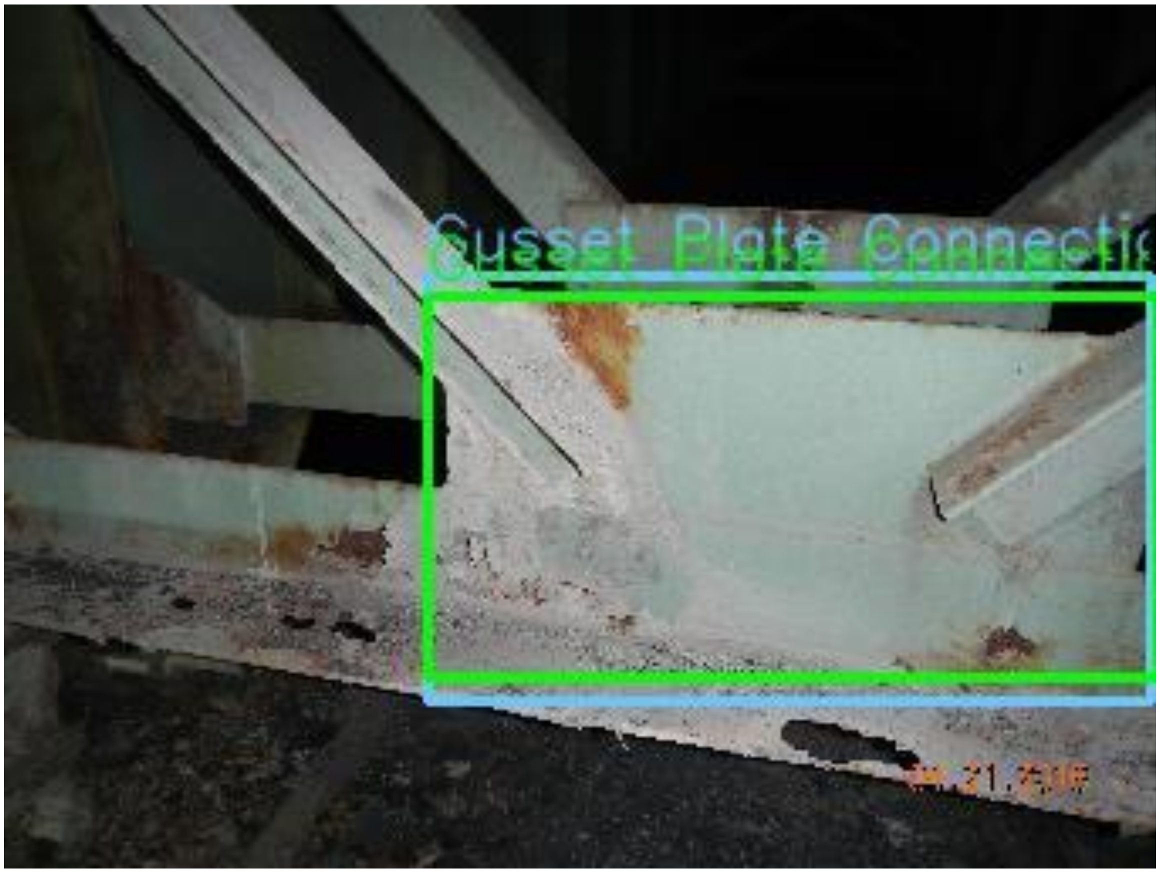


Figure D.39

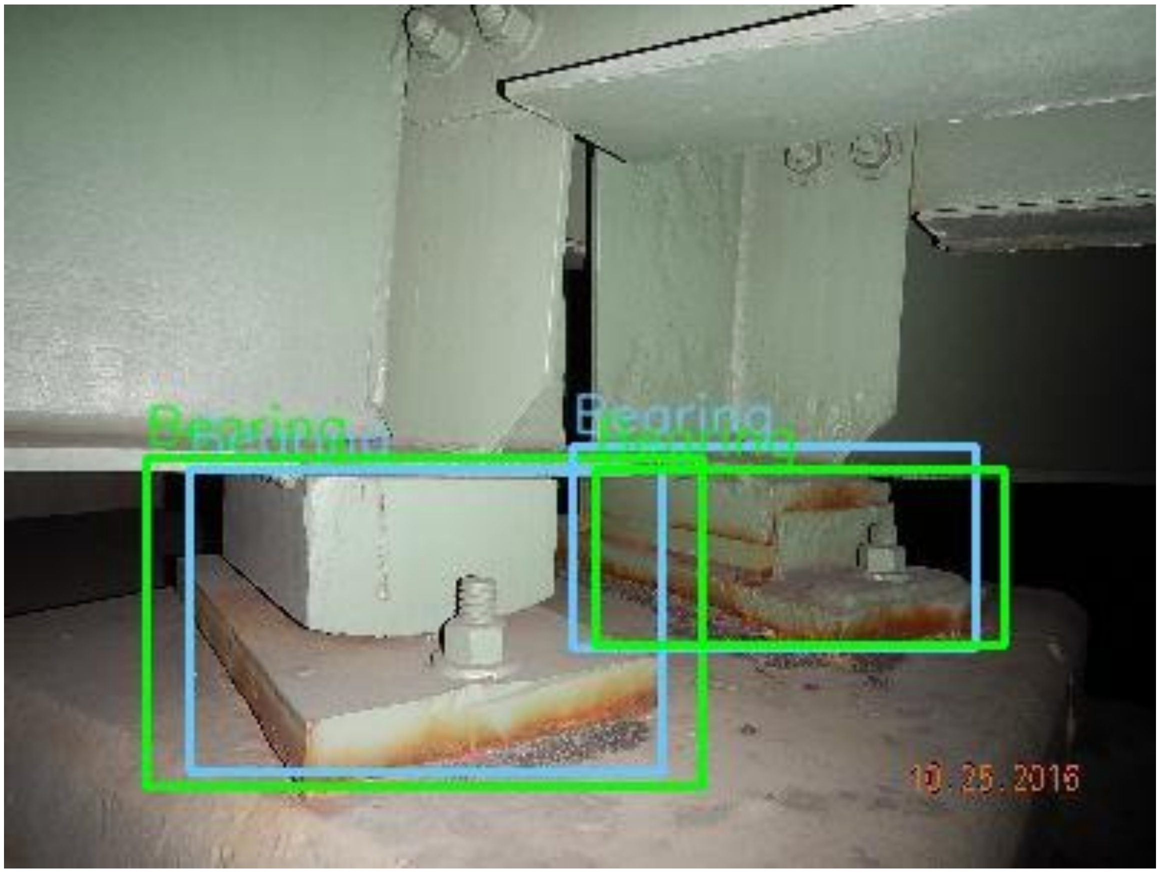


Figure D.40

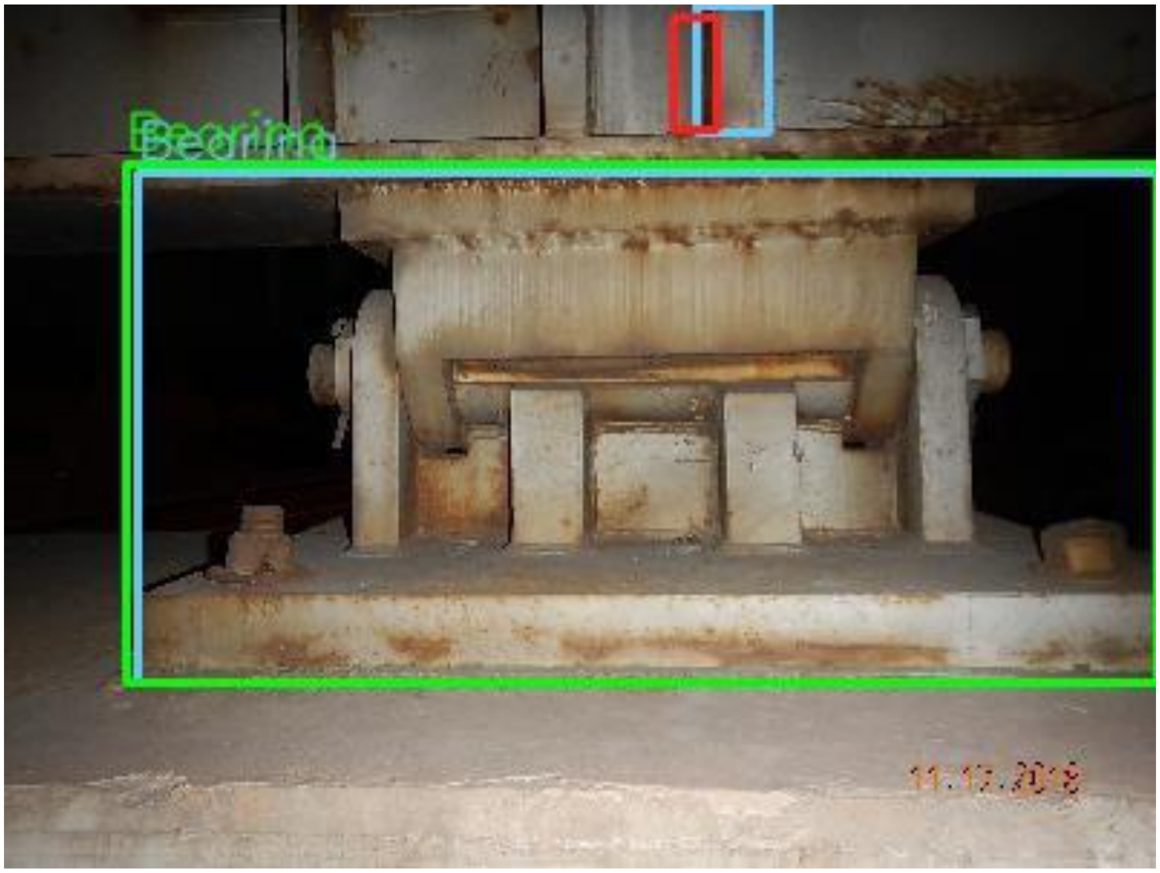


Figure D.41

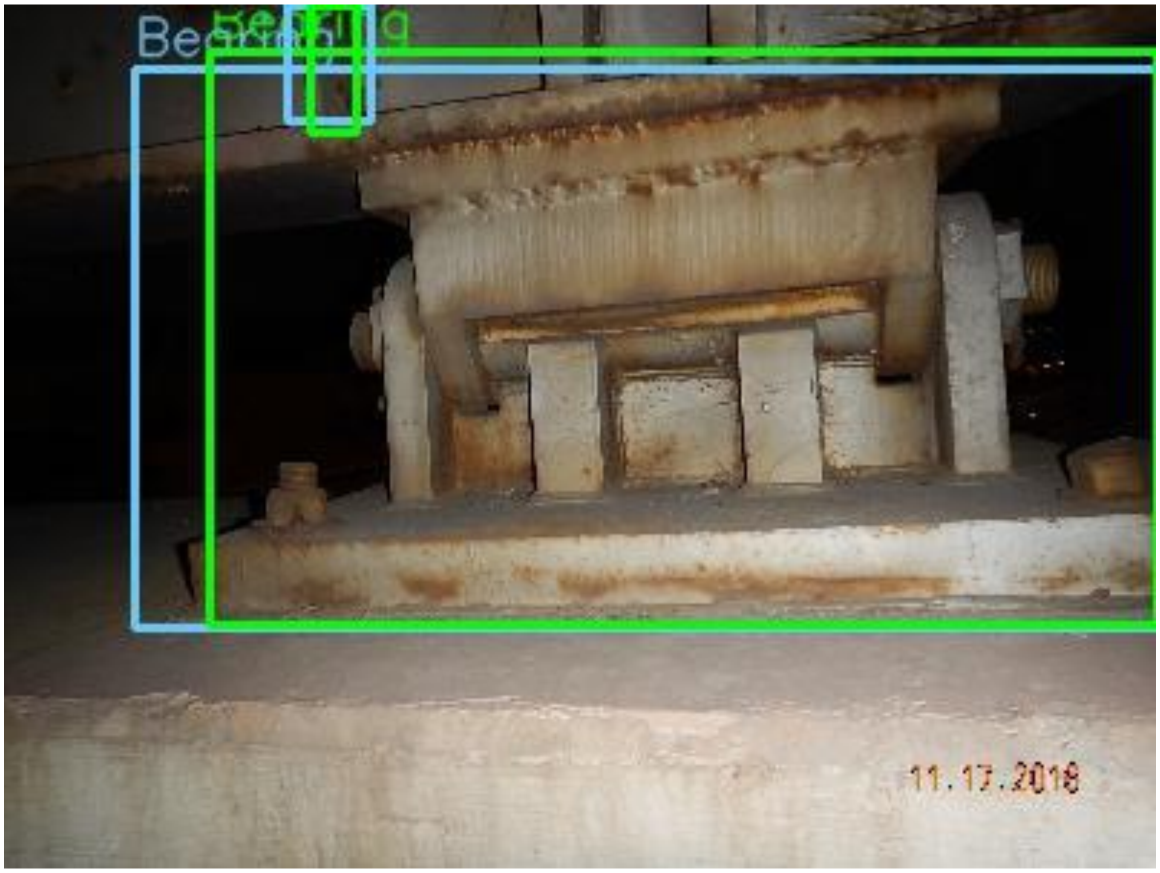
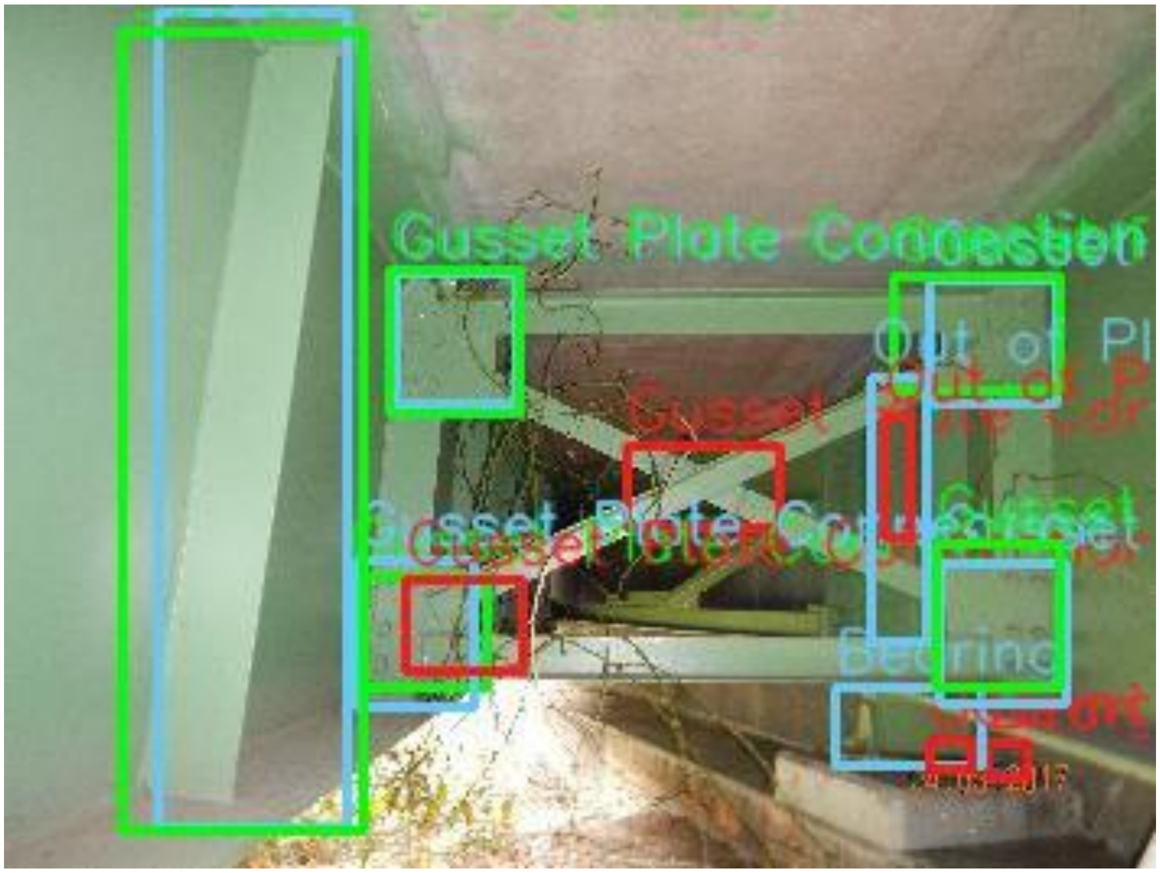


Figure D.42


Gusset Plate Connection
Out of Pl
Gusset
Gusset Plate Connection
Gusset Plate Connection
Gusset
Bearing
Figure D.43

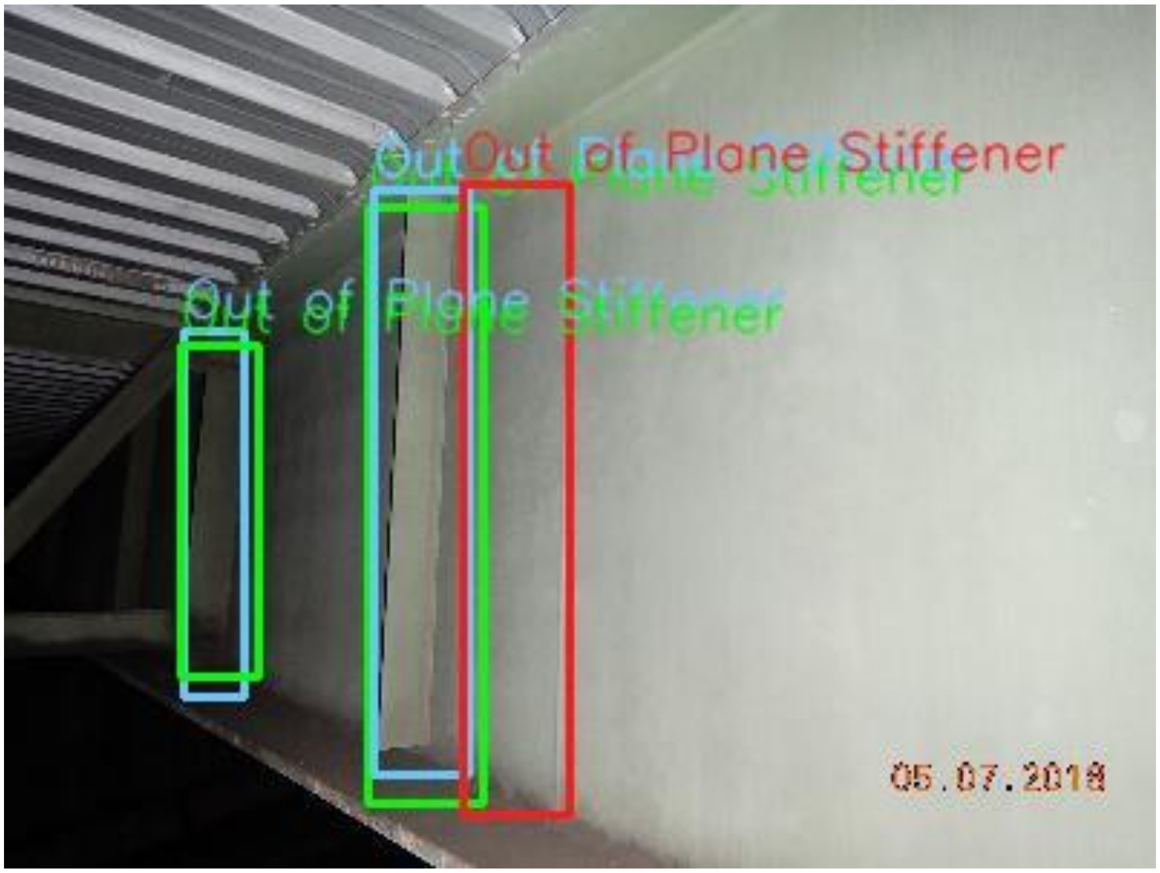

Out of Plane Stiffener
Out of Plane Stiffener
Out of Plane Stiffener
Out of Plane Stiffener
Out of Plane Stiffener
05.07.2018

Figure D.44

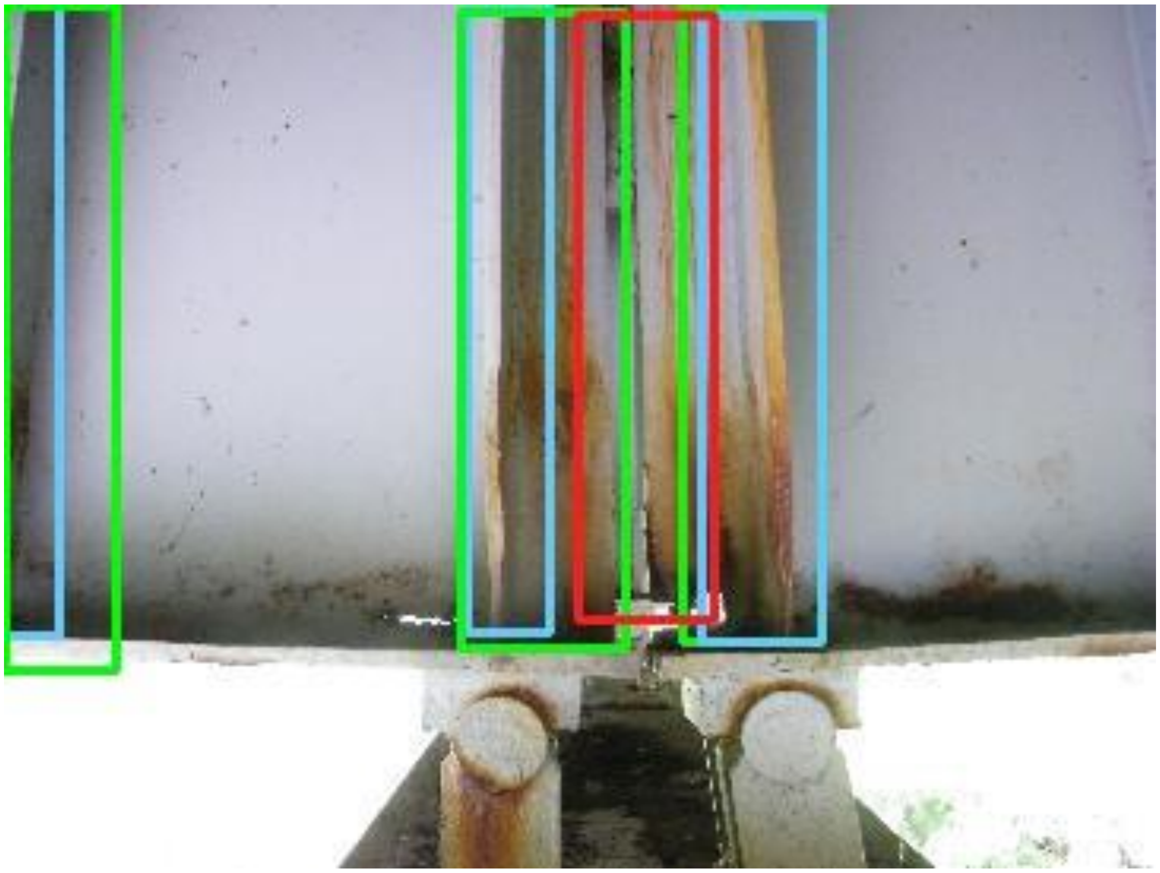


Figure D.45

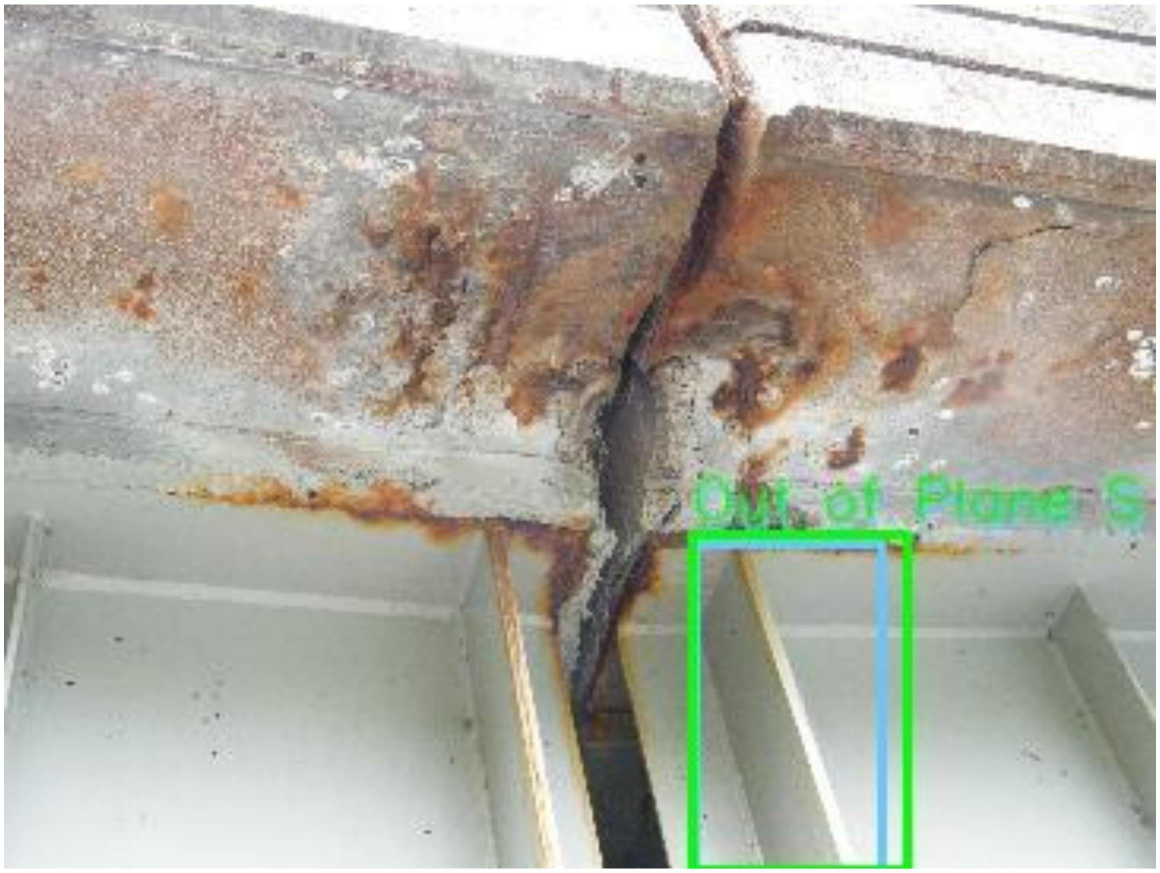


Figure D.46

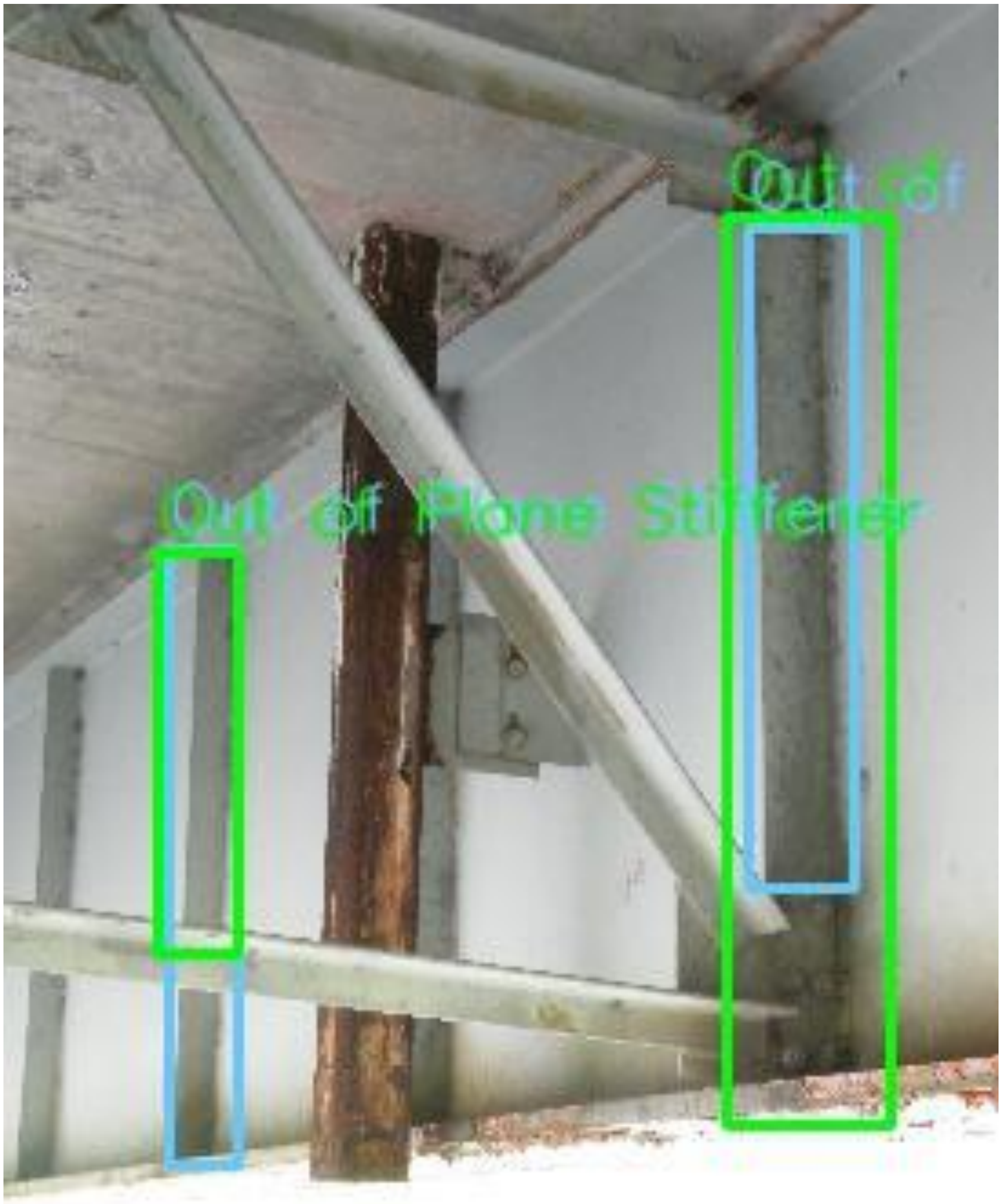


Figure D.47

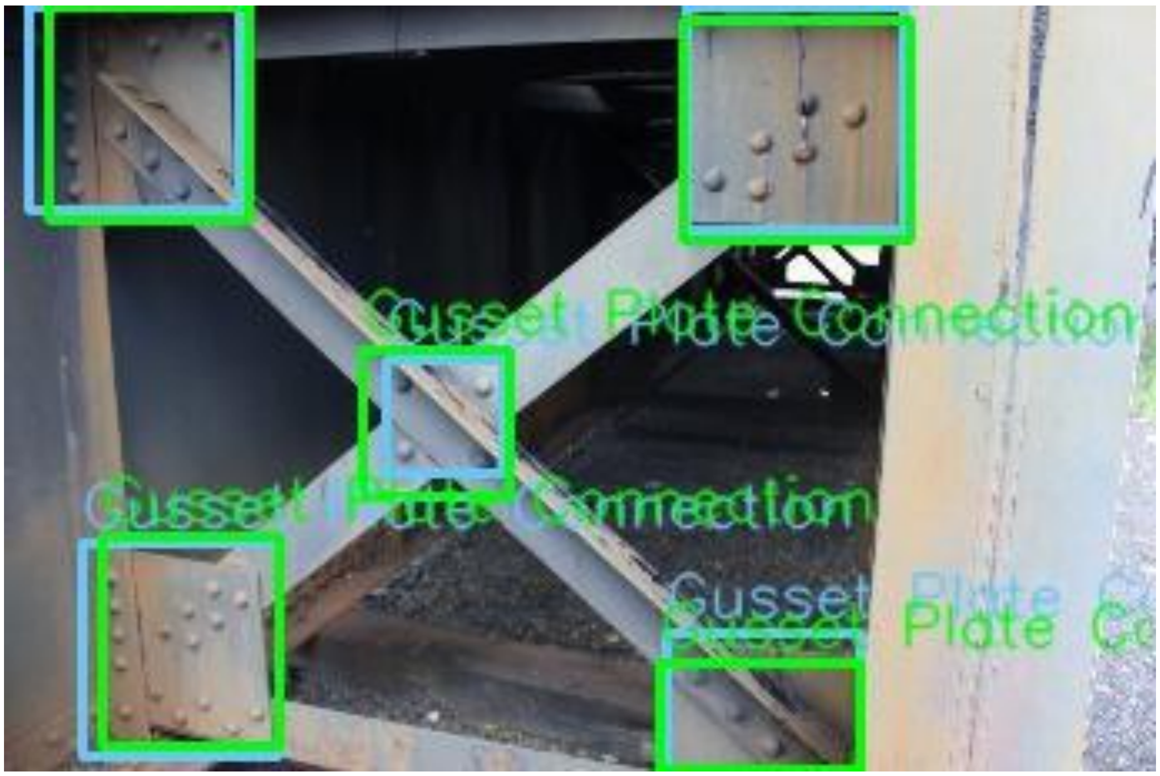


Figure D.48

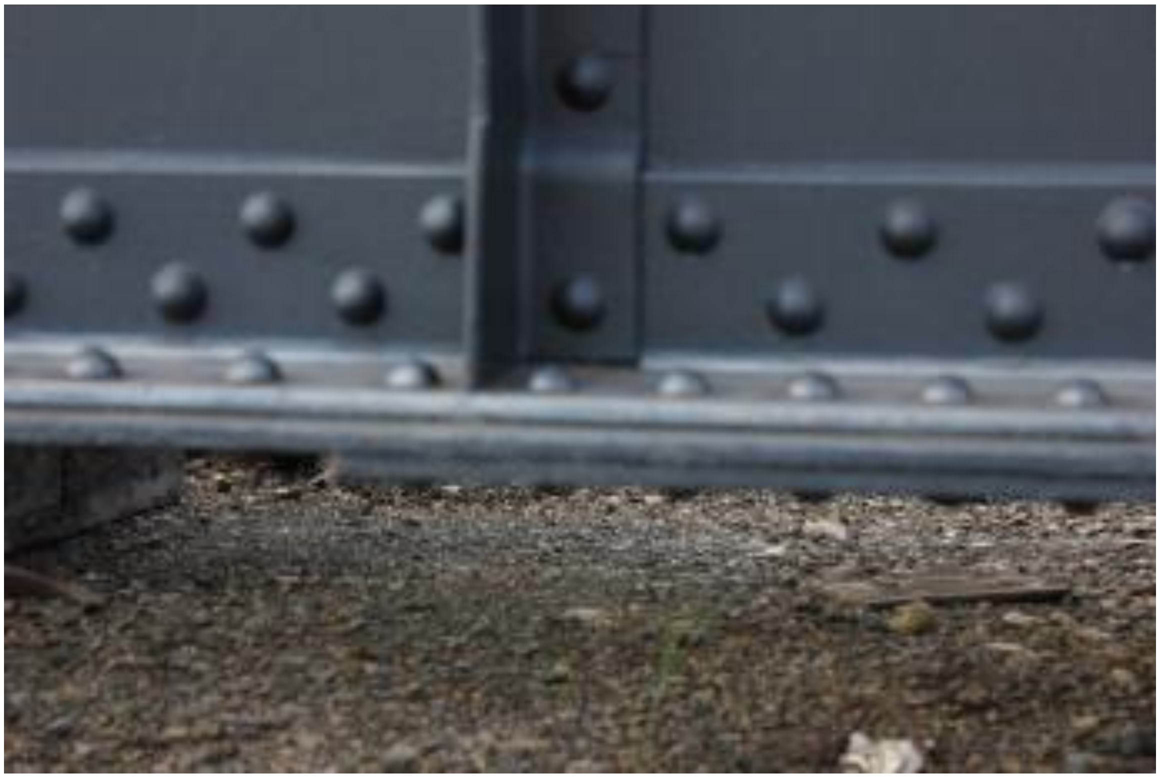


Figure D.49 (No Prediction)

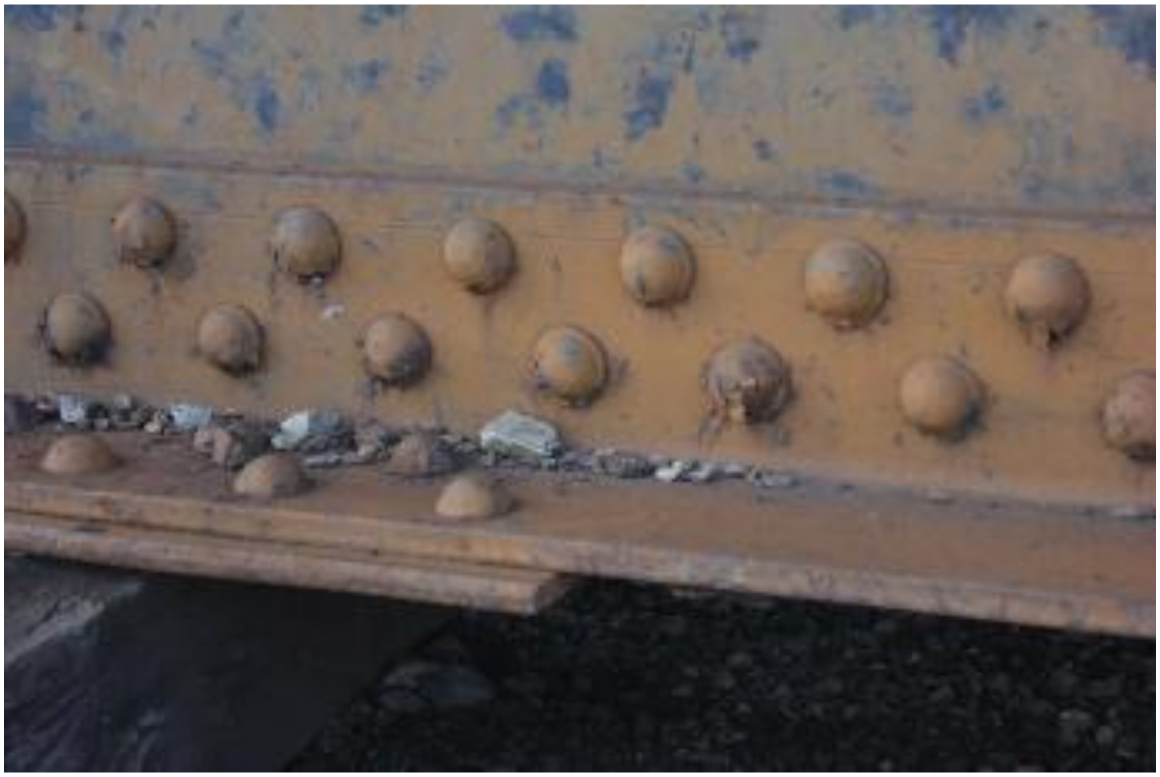


Figure D.50 (No Prediction)

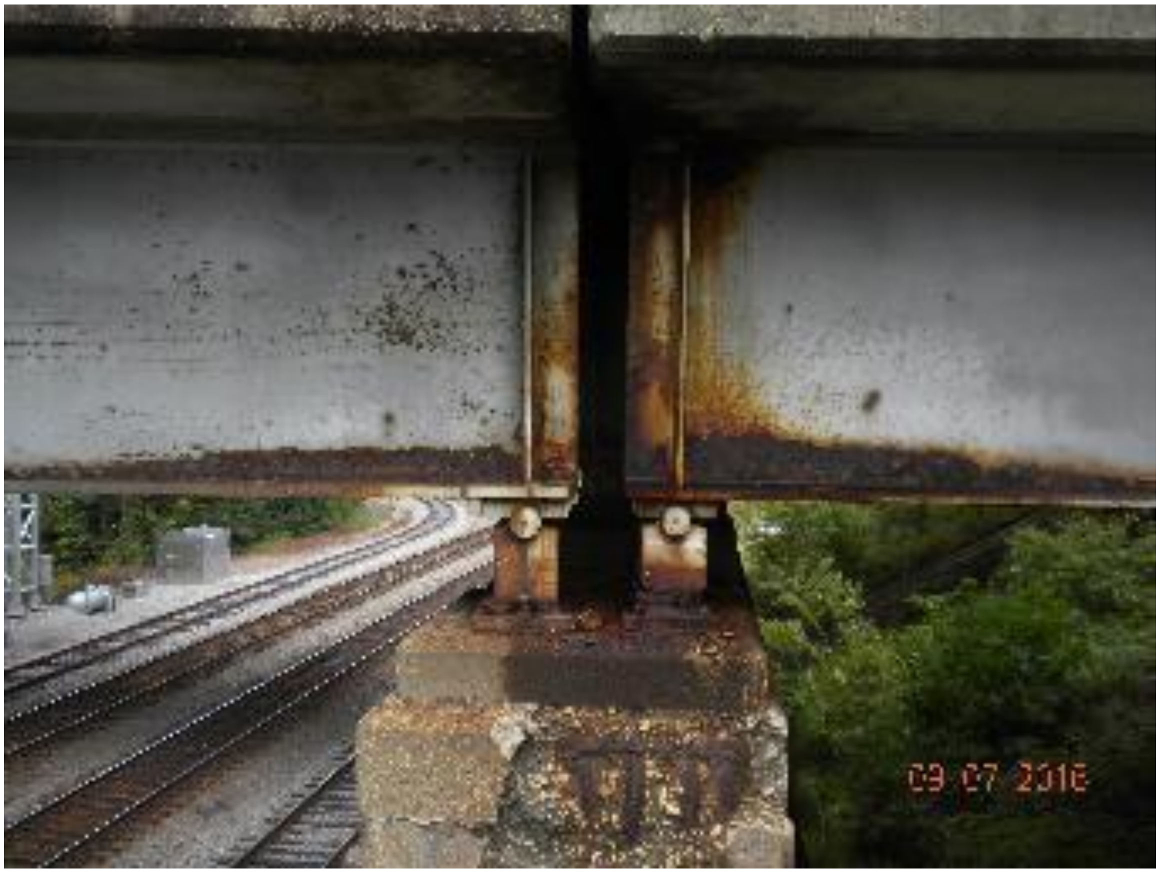


Figure D.51 (No Prediction)

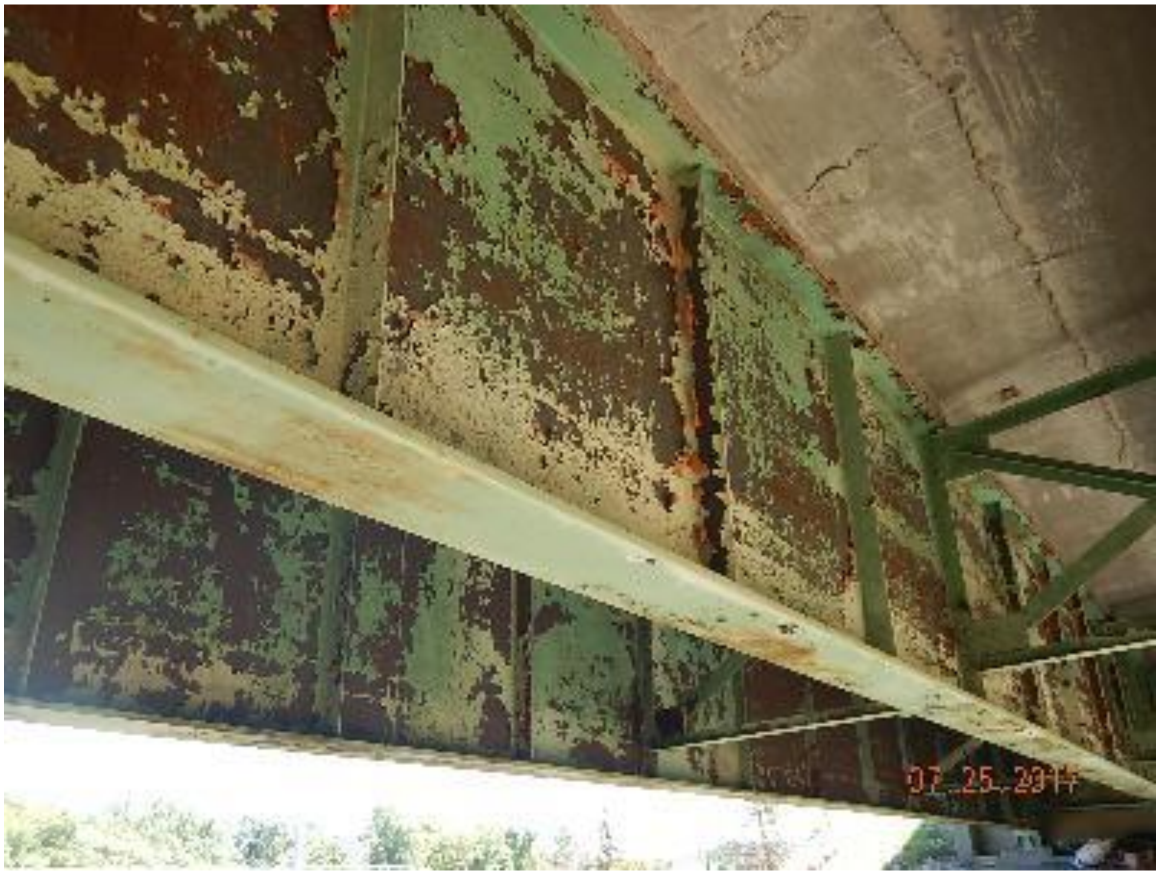


Figure D.52 (No Prediction)

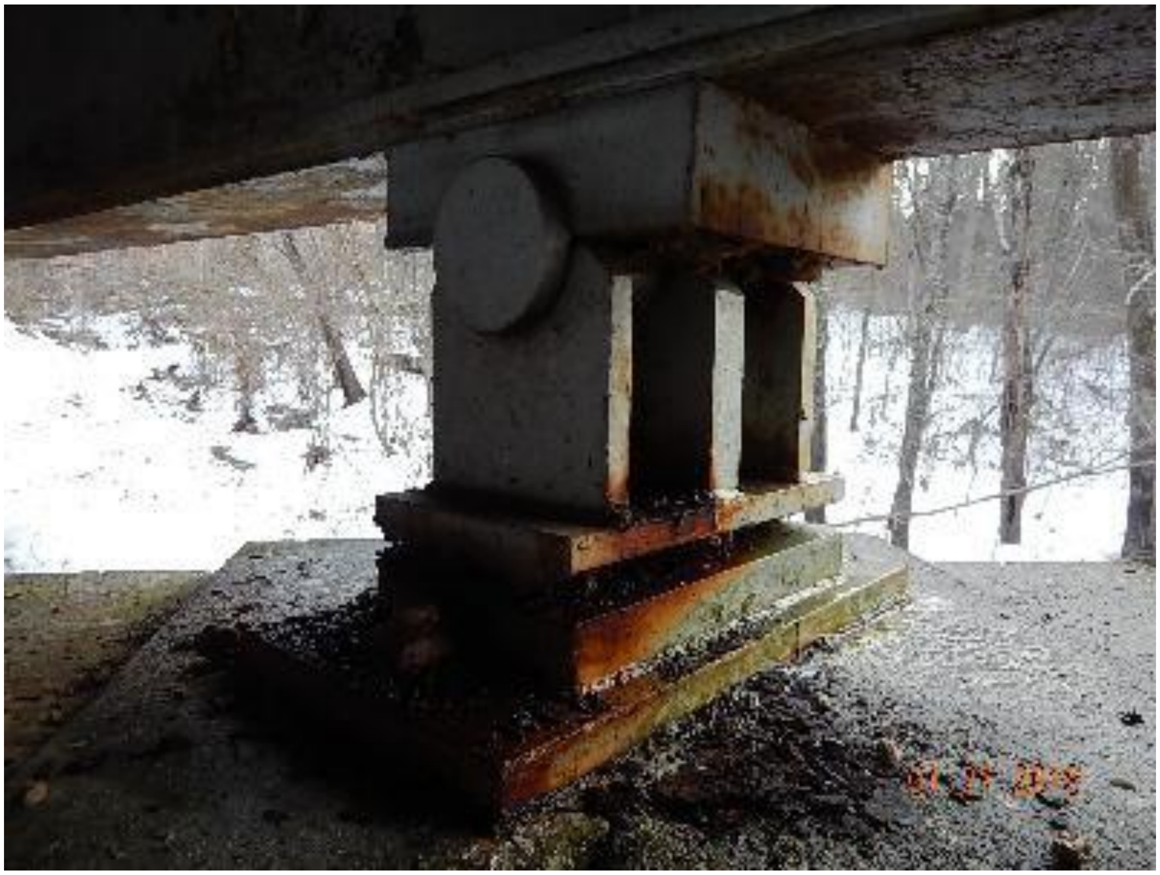


Figure D.53 (No Prediction)

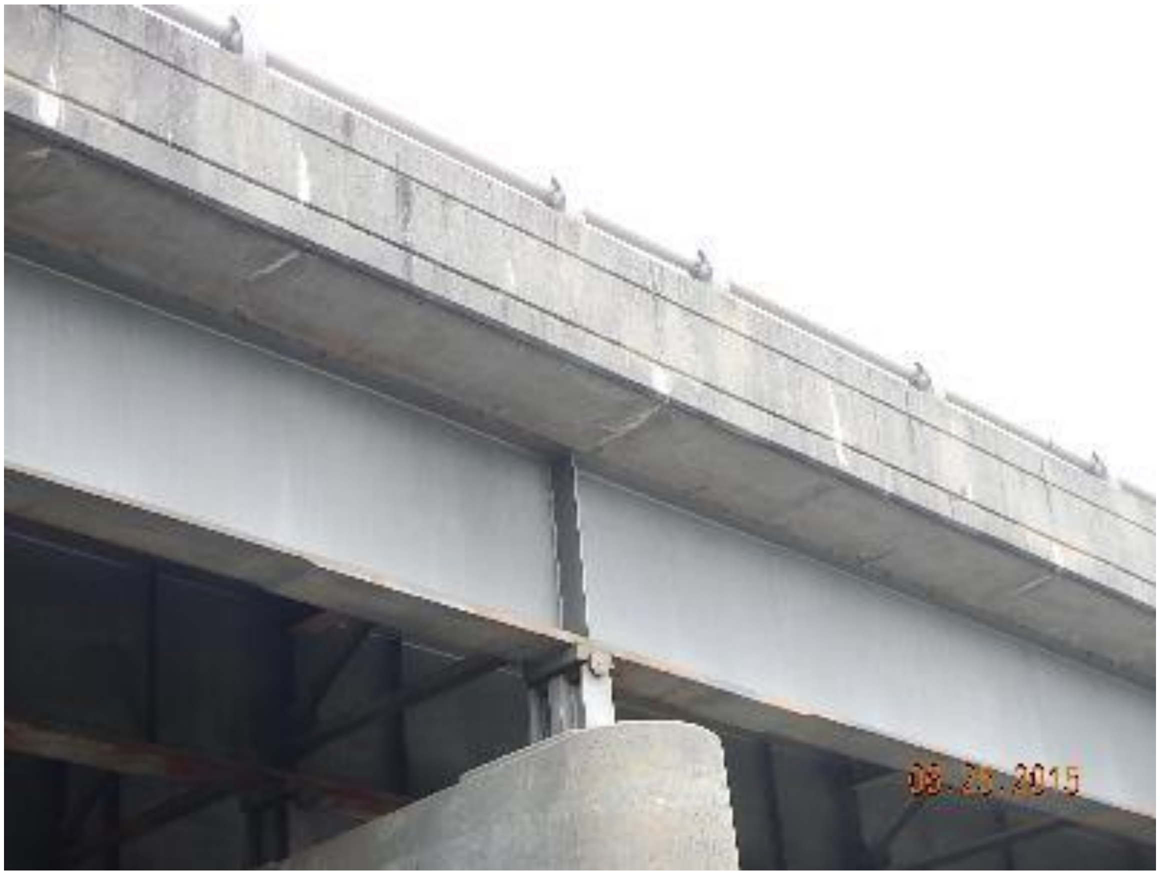


Figure D.54 (No Prediction)

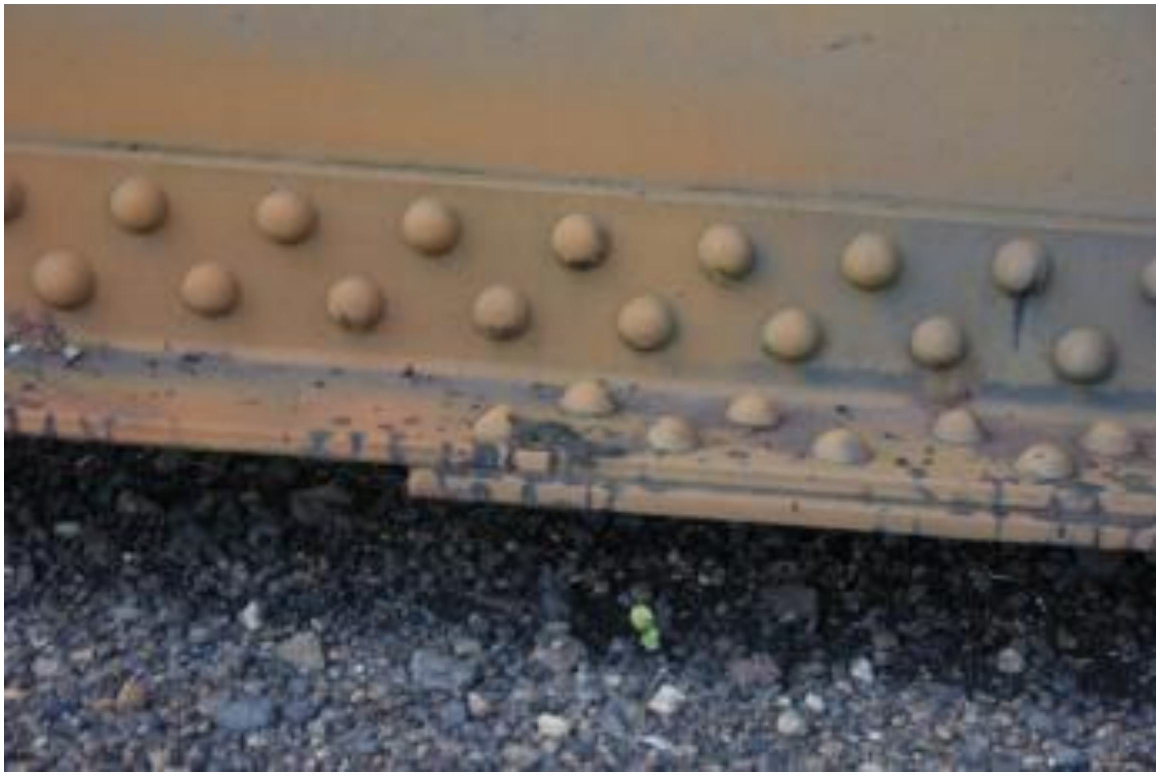


Figure D.55 (No Prediction)

APPENDIX E – TENSORFLOW ERROR LOSS VISUALIZATION

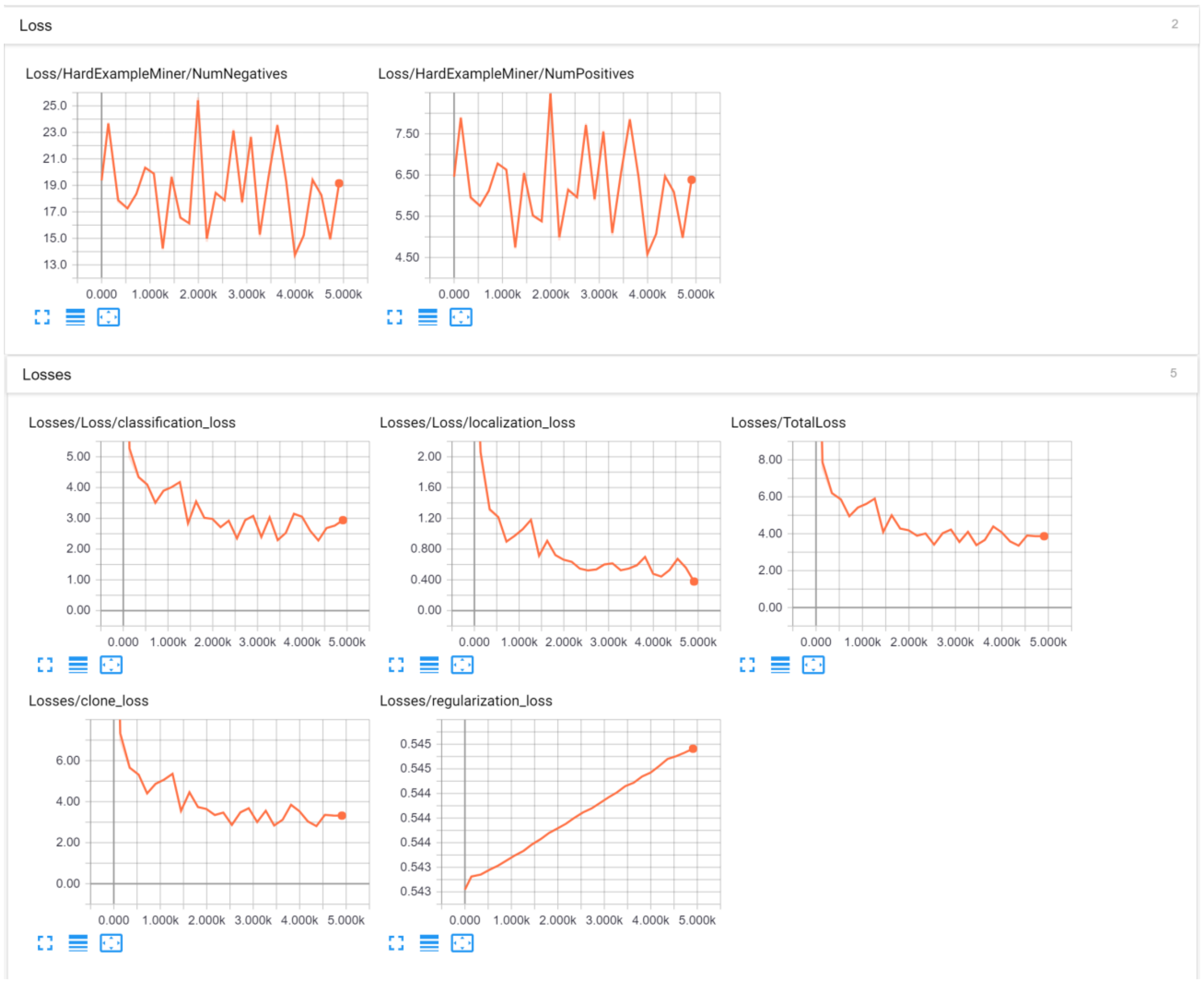


Figure E.1 - Losses for Model (A_T)

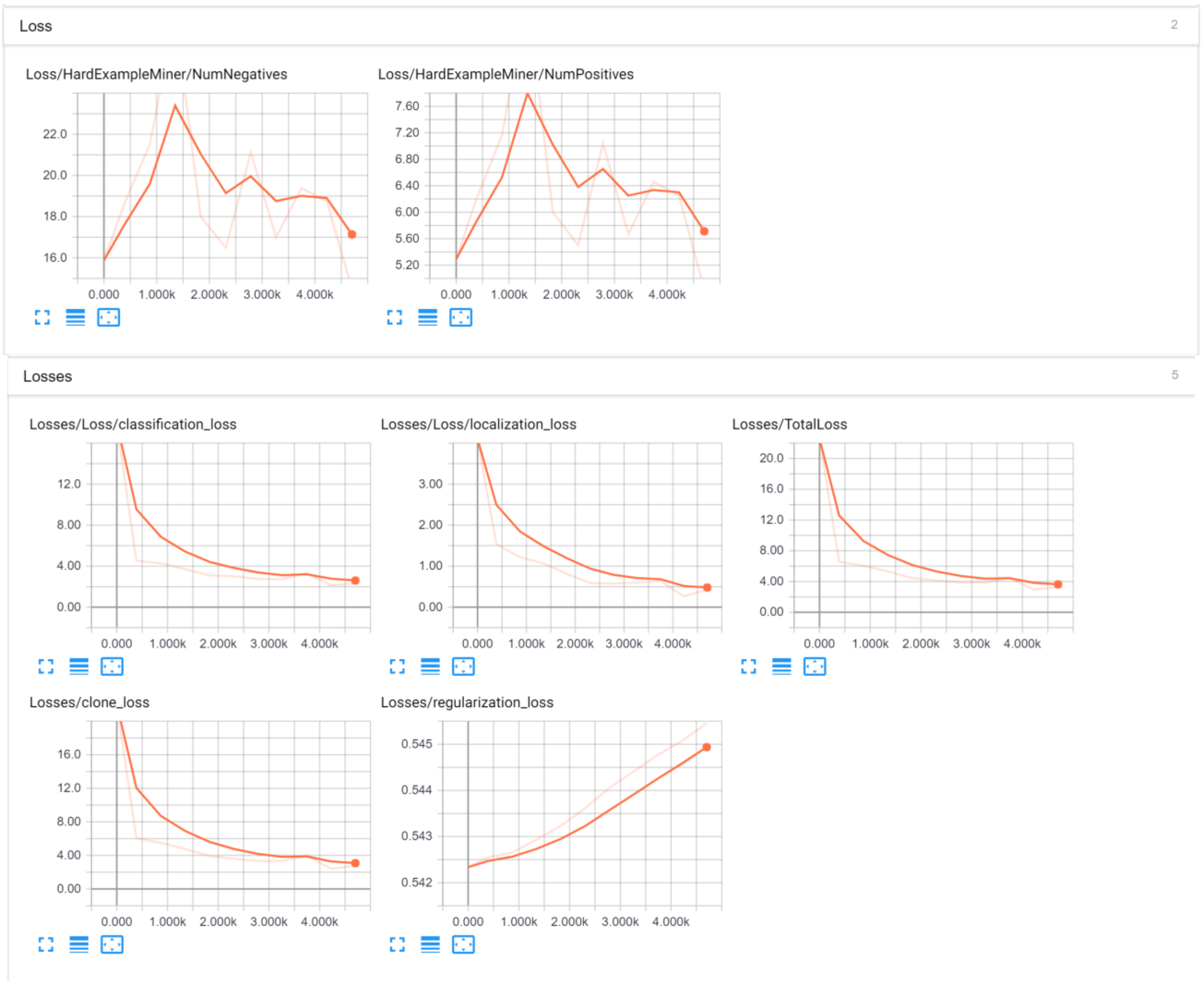


Figure E.2 – Losses for Model (B_T)

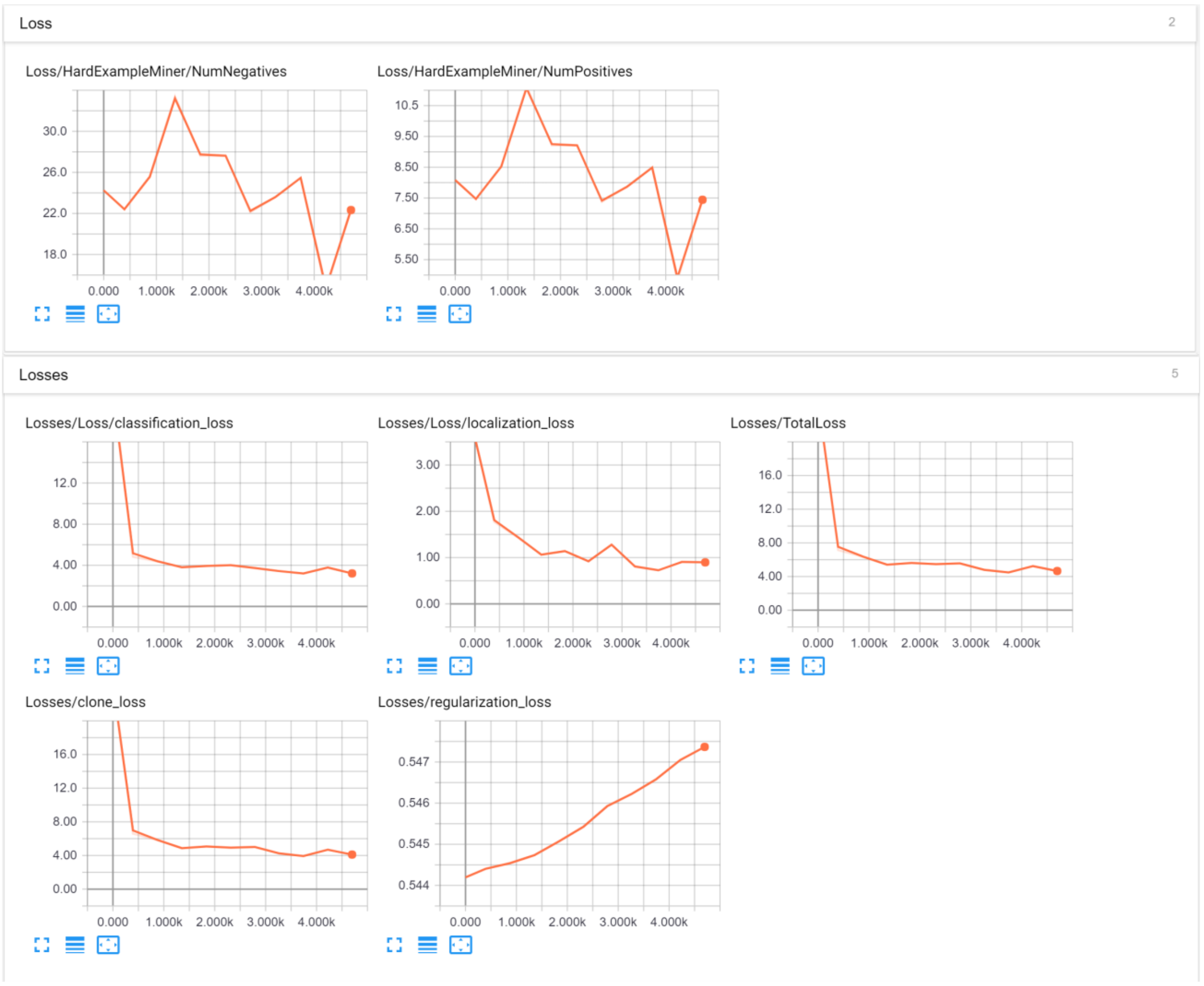


Figure E.3 – Losses for Model (C_T)

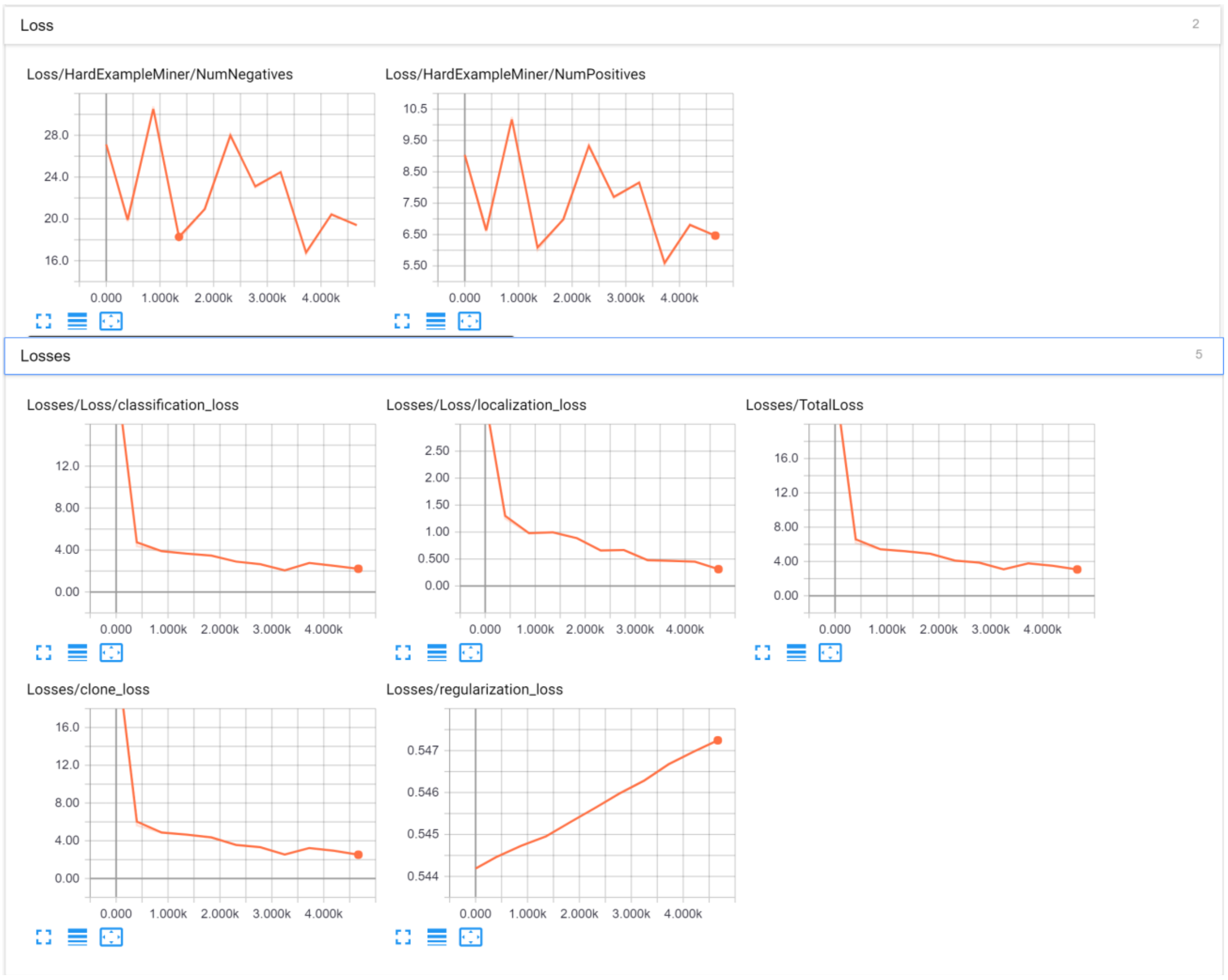


Figure E.4 – Losses for Model (D_T)

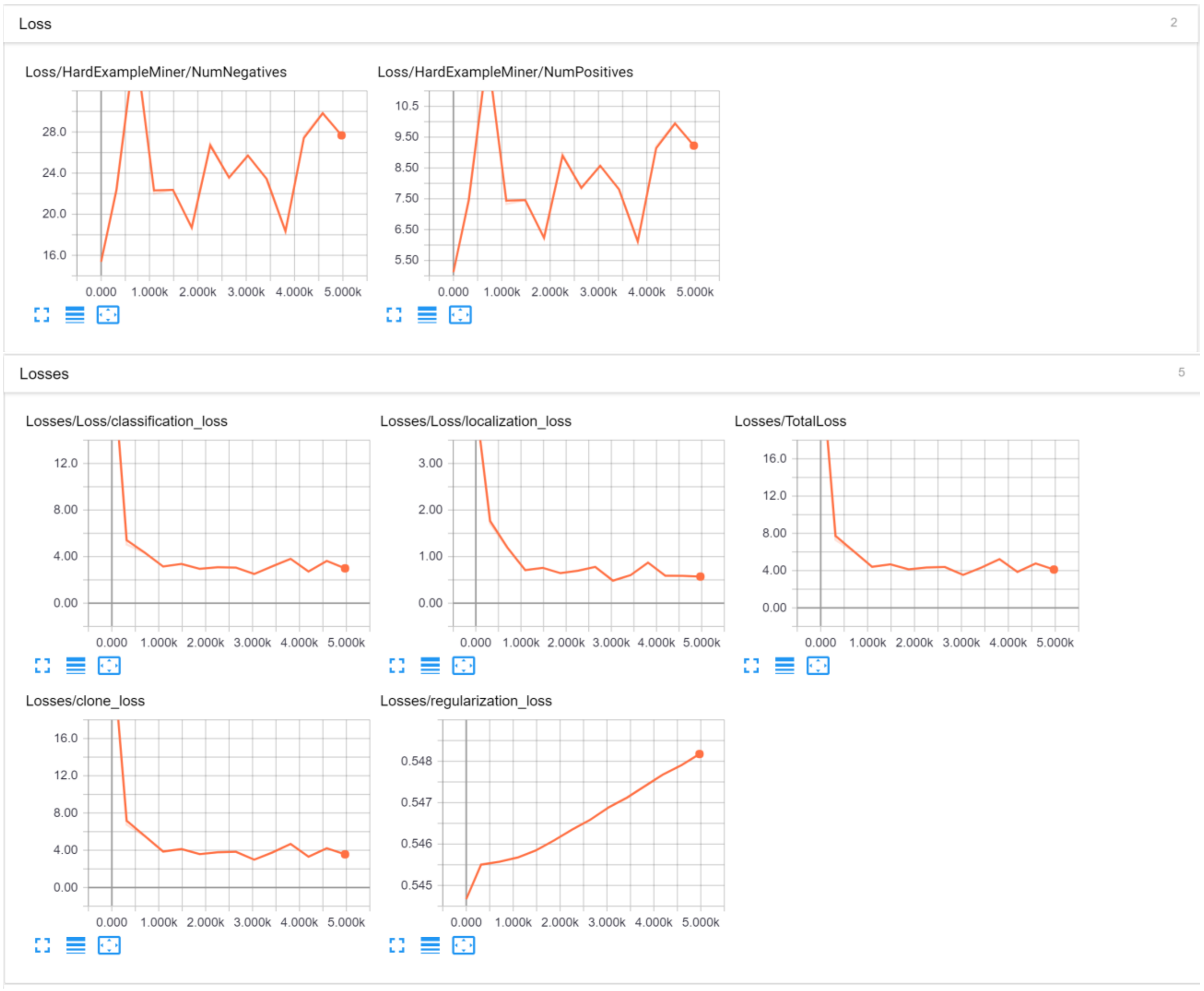


Figure E.5 – Losses for Model (E_T)

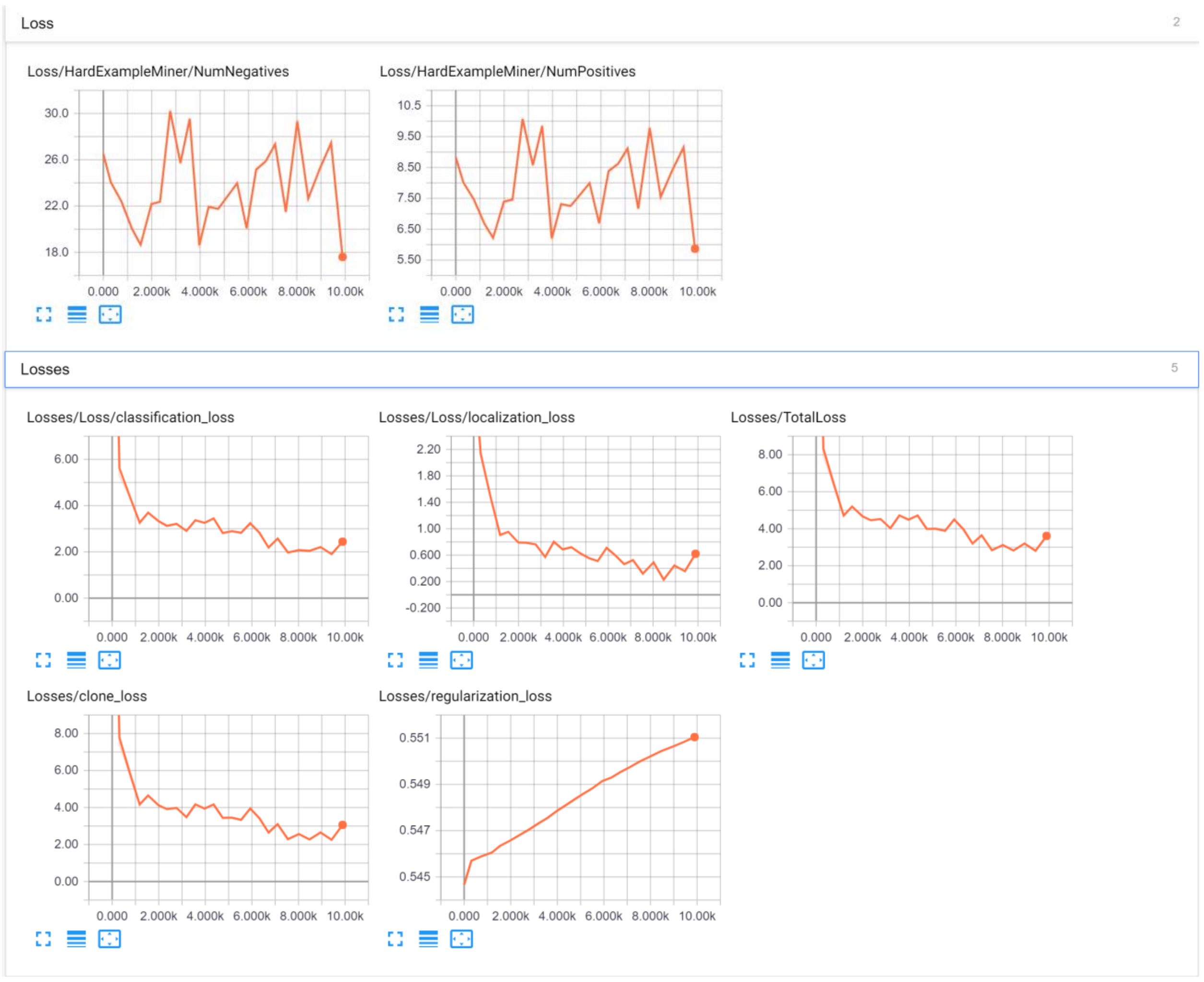
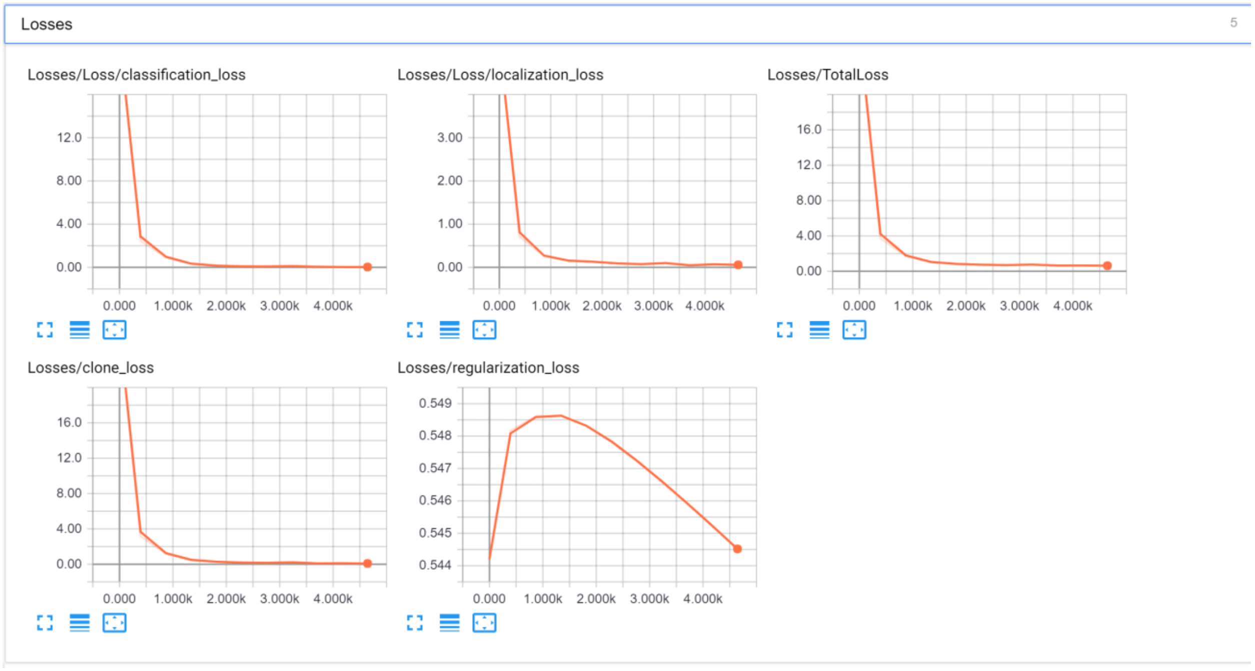


Figure E.6 – Losses for Model (E_T*)



Losses
5
Losses/Loss/classification_loss
Losses/Loss/localization_loss
Losses/TotalLoss
Losses/clone_loss
Losses/regularization_loss
0.000
1.000k
2.000k
3.000k
4.000k

Figure E.7 – De-bugging Example: Unrealistic Loss Curve Behavior (1)

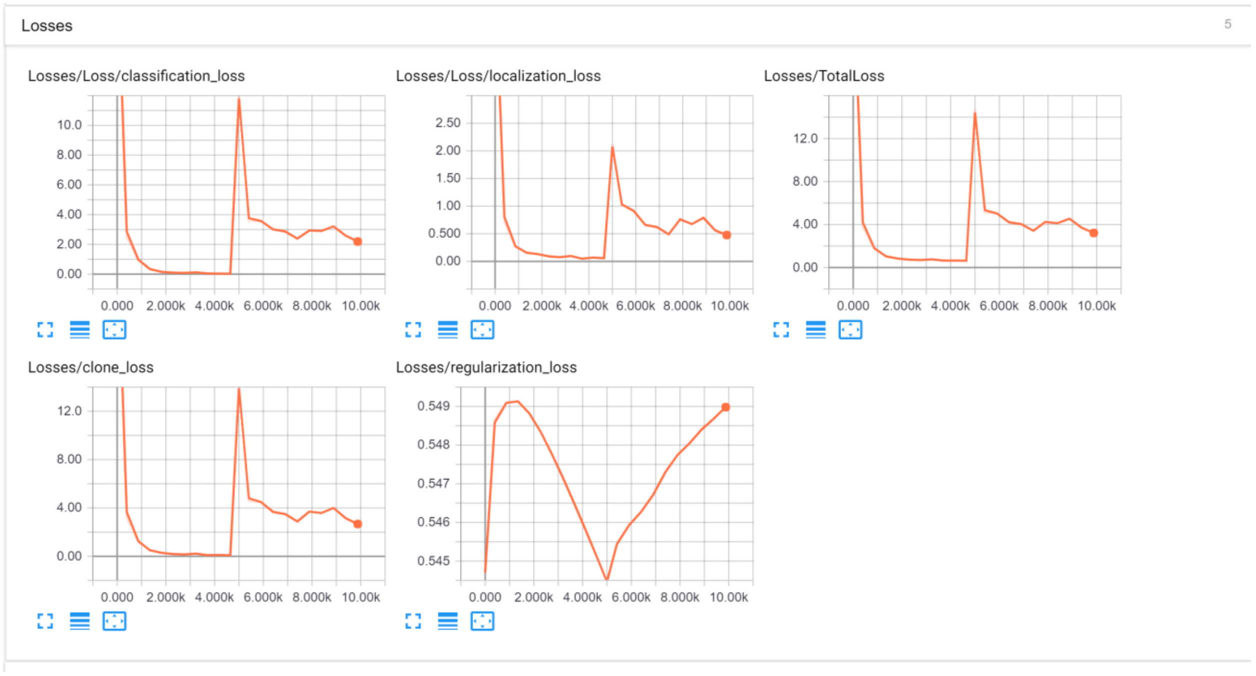


Figure E.8 – De-bugging Example: Unrealistic Loss Curve Behavior (2)

REFERENCES

Abu-Tair, A., Mcparland, C., Lyness, J., & Nadjai, A. (2002). Predictive Models Of Deterioration Rates Of Concrete Bridges Using The Factor Method Based On Historic Inspection Data. Retrieved from https://www.irbnet.de/daten/iconda/CIB10090.pdf

Ankit Gupta. (2017). How to read most commonly used file formats in Data Science (using Python)? Retrieved June 14, 2018, from https://www.analyticsvidhya.com/blog/2017/03/read-commonly-used-formats-using-python/

ASCE. (2017). *2017 Infrastructure Report Card: Bridges*. Retrieved from https://www.infrastructurereportcard.org/wp-content/uploads/2017/01/Bridges-Final.pdf

Bagnard, M. (2016). *Gusset Plate Inspection Issues*. Retrieved from https://www.ntsb.gov/news/events/Documents/minneapolis_mn-9_Gusset_Plate_Inspection_Issues.pdf

Brett. (2016). What Exactly is a Pixel Perfect Design? | Brett Dev. Retrieved June 8, 2018, from https://www.brettdev.com/exactly-pixel-perfect-design/

Bridge Bearings. (2018). Retrieved November 20, 2018, from http://www.bridgedesign.org.uk/parts/bearing.html

Cha, Y.-J., Choi, W., & Büyüköztürk, O. (2017). Deep Learning-Based Crack Damage Detection Using Convolutional Neural Networks. *Computer-Aided Civil and Infrastructure Engineering*, *32*(5), 361–378. https://doi.org/10.1111/mice.12263

Chordia, S., & Verma, R. (2015). Plankton Image Classification. Retrieved from http://cs231n.stanford.edu/reports/2015/pdfs/sagarc14_final_report.pdf

Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., … Darmstadt, T. U. (2016). *The Cityscapes Dataset for Semantic Urban Scene Understanding*. Retrieved from www.cityscapes-dataset.net

Deng, J., Russakovsky, O., Krause, J., Bernstein, M., Berg, A., & Fei-Fei, L. (2014). Scalable Multi-label Annotation. https://doi.org/10.1145/2556288.2557011

Everingham, M., Gool, L. Van, Williams, C. K. I., & Winn, J. (2010). The PASCAL Visual Object Classes (VOC) Challenge. *International Journal*, 303–338. https://doi.org/10.1007/s11263-009-0275-4

FAA. (2018). Fact Sheet – Small Unmanned Aircraft Regulations (Part 107). Retrieved June 9, 2018, from https://www.faa.gov/news/fact_sheets/news_story.cfm?newsId=20516

Gillins, D; Parrish, C; Gillins, M; Simpson, C. (2018). *EYES IN THE SKY: BRIDGE INSPECTIONS WITH UNMANNED AERIAL VEHICLES*. Retrieved from https://www.oregon.gov/ODOT/Programs/ResearchDocuments/SPR787_Eyes_in_the_Sky.pdf

Girshick, R. (2015). Fast R-CNN. In *Proceedings of the IEEE International Conference on Computer Vision*. https://doi.org/10.1109/ICCV.2015.169

Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. Retrieved from http://www.cs.berkeley.edu

Guldur, B., & Hajjar, J. F. (2016). Automated Classification of Detected Surface Damage from Point

Clouds with Supervised Learning. Retrieved from http://www.iaarc.org/publications/fulltext/ISARC2016-Paper057.pdf

Haghani, R., Al-Emrani, M., & Heshmati, M. (2012). Fatigue-Prone Details in Steel Bridges. *Buildings*, *2*(4), 456–476. https://doi.org/10.3390/buildings2040456

Hallermann, N., Morgenthal, G., & Rodehorst, V. (2015). Unmanned Aerial Systems (UAS) – Case Studies of Vision Based Monitoring of Ageing Structures. *International Symposium Non-Destructive Testing in Civil Engineering (NDT-CE)*, (November).

He, K., Gkioxari, G., Dollar, P., & Girshick, R. (2017). Mask R-CNN. In *Proceedings of the IEEE International Conference on Computer Vision*. https://doi.org/10.1109/ICCV.2017.322

Hoskere, V., Narazaki, Y., Hoang, T. A., & Spencer, B. F. (2017). Vision-based Structural Inspection using Multiscale Deep Convolutional Neural Networks. Retrieved from https://arxiv.org/ftp/arxiv/papers/1805/1805.01055.pdf

Hu, H., Gu, J., Zhang, Z., Dai, J., & Wei, Y. (2017). *Relation Networks for Object Detection*. Retrieved from http://openaccess.thecvf.com/content_cvpr_2018/papers/Hu_Relation_Networks_for_CVPR_2018_paper.pdf

Hüthwohl, P., Lu, R., & Brilakis, I. (2016). Challenges of bridge maintenance inspection. *Icccbe 2016*, (July).

Ibrahim, A., Abbott, A. L., & Hussein, M. E. (2016). An Image Dataset of Text Patches in Everyday Scenes. Retrieved from https://arxiv.org/pdf/1610.06494.pdf

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet Classification with Deep Convolutional Neural Networks. Retrieved from https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf

LeCun, Y., Cortes, C., & Burges, C. (1998). MNIST handwritten digit database, Yann LeCun, Corinna Cortes and Chris Burges. Retrieved June 10, 2018, from http://yann.lecun.com/exdb/mnist/

Lin, T.-Y., Maire, M., Belongie, S., Bourdev, L., Girshick, R., Hays, J., … Dolí, P. (2016). Microsoft COCO: Common Objects in Context. Retrieved from https://arxiv.org/pdf/1405.0312.pdf

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., & Berg, A. C. (2016). *SSD: Single Shot MultiBox Detector*. Retrieved from https://github.com/weiliu89/caffe/tree/ssd

Lovelace, B. (2015). Unmanned Aerial Vehicle Bridge Inspection Demonstration Project. Retrieved from http://www.dot.state.mn.us/research/TS/2015/201540.pdf

Mohan, A., & Poobal, S. (2017). Crack detection using image processing: A critical review and analysis. *Alexandria Engineering Journal*. https://doi.org/10.1016/J.AEJ.2017.01.020

ODOT. (1973). *Ohio Department of Transportation Manual of Bridge Inspection*. Retrieved from http://www.dot.state.oh.us/Divisions/Engineering/Structures/bridge operations and maintenance/Manual of Bridge Inspection 2014 v8/Manual of Bridge Inspection 2014 v8 without Appendix.pdf

Otero, L. D. (2013). Proof of Concept for using Unmanned Aerial Vehicles for High Mast Pole and Bridge Inspections, *2015*(June 2013). https://doi.org/10.13140/RG.2.1.4567.2166

Prasanna, P., Dana, K. J., Gucunski, N., Basily, B. B., La, H. M., Lim, R. S., … Basily, B. (2016). Automated Crack Detection on Concrete Bridges. *IEEE TRANSACTIONS ON AUTOMATION SCIENCE AND ENGINEERING, 13*(2). https://doi.org/10.1109/TASE.2014.2354314

Rau, J. Y., Hsiao, K. W., Jhan, J. P., Wang, S. H., Fang, W. C., & Wang, J. L. (2016). BRIDGE CRACK DETECTION USING MULTI-ROTARY UAV AND OBJECT-BASE IMAGE ANALYSIS. Retrieved from http://uavg17.ipb.uni-bonn.de/wp-content/papercite-data/pdf/rau2017uavg-54.pdf

Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2015). *You Only Look Once: Unified, Real-Time Object Detection*. Retrieved from http://pjreddie.com/yolo/

Ren, S., He, K., Girshick, R., & Sun, J. (2016). Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. Retrieved from https://arxiv.org/pdf/1506.01497.pdf

Rosebrock, A. (2016). Intersection over Union (IoU) for object detection - PyImageSearch. Retrieved November 21, 2018, from https://www.pyimagesearch.com/2016/11/07/intersection-over-union-iou-for-object-detection/

Russell, B. C., Torralba, A., Murphy, K. P., & Freeman, W. T. (2008). LabelMe: a database and web-based tool for image annotation. *INTERNATIONAL JOURNAL OF COMPUTER VISION*, *77*(3), 157–173. Retrieved from http://people.csail.mit.edu/brussell/research/AIM-2005-025-new.pdf

Shen, Y., Lindenbergh, R., & Wang, J. (2017). Change analysis in structural laser scanning point clouds: The baseline method. *Sensors (Switzerland)*, *17*(1), 1–25. https://doi.org/10.3390/s17010026

Siam, M., Elkerdawy, S., Jagersand, M., & Yogamani, S. (2017). Deep Semantic Segmentation for Automated Driving: Taxonomy, Roadmap and Challenges. Retrieved from https://arxiv.org/pdf/1707.02432.pdf

Singh, D., & Garzon, P. (2016). *Using Convolutional Neural Networks and Transfer Learning to Perform Yelp Restaurant Photo Classification*. Retrieved from http://cs231n.stanford.edu/reports/2016/pdfs/001_Report.pdf

Skymind. (2017). AI vs. ML vs. DL. Retrieved June 9, 2018, from https://deeplearning4j.org/ai-machinelearning-deeplearning

Sonnenberg, A. (n.d.). Predictive Modeling of Bridges. Retrieved from https://www.eiseverywhere.com/file_uploads/bb63e1f34401f34905da00d1f36e6245_PredictiveModelingofBridges.pdf

Stanford. (2013). Feature extraction using convolution - Ufldl. Retrieved December 18, 2018, from http://deeplearning.stanford.edu/wiki/index.php/Feature_extraction_using_convolution

Stent, S., Gherardi, R., Stenger, B., & Cipolla, R. (2015). Detecting Change for Multi-View, Long-Term Surface Inspection. Retrieved from http://mi.eng.cam.ac.uk/~cipolla/publications/inproceedings/2015-BMVC-change-detection.pdf

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., … Rabinovich, A. (2014). *Going Deeper with Convolutions*. Retrieved from https://www.cs.unc.edu/~wliu/papers/GoogLeNet.pdf

U.S. Department of Transportation. (2012). *Steel Bridge Design Handbook Design for Fatigue*. Retrieved from https://www.fhwa.dot.gov/bridge/steel/pubs/if12052/volume12.pdf

University of Wisconsin. (2007). A Basic Introduction To Neural Networks. Retrieved June 9, 2018, from http://pages.cs.wisc.edu/~bolo/shipyard/neural/local.html

Veit, A., Matera, T., Neumann, L., Matas, J., & Belongie, S. (2016a). COCO-Text: Dataset and Benchmark for Text Detection and Recognition in Natural Images. Retrieved from https://vision.cornell.edu/se3/wp-content/uploads/2016/01/1601.07140v1.pdf

Veit, A., Matera, T., Neumann, L., Matas, J., & Belongie, S. (2016b). COCO-Text: Dataset and Benchmark for Text Detection and Recognition in Natural Images. Retrieved from https://vision.cornell.edu/se3/wp-content/uploads/2016/01/1601.07140v1.pdf

Wang, K. C. P., Allen, T., Yang, A., Cheng, J., & Gary, G. (2017). Deep Learning System for Automated Cracking Survey &amp; Its Performance with Pixel Accuracy: CrackNet. Retrieved from http://www.rpug.org/download/2017/6-2-Kelvin Wang.pdf

Yeum, C. M. (2016). COMPUTER VISION-BASED STRUCTURAL ASSESSMENT EXPLOITING LARGE VOLUMES OF IMAGES. Retrieved from https://engineering.purdue.edu/IISL/Publications/DSc_Dissertations/Chul_Min_Yeum.pdf

Zheng, P., Moen, C., Roberts-Wollmann, C., & Cousins, T. (2014). Crack Detection and Measurement Utilizing Image-Based Reconstruction. Retrieved from https://vtechworks.lib.vt.edu/bitstream/handle/10919/48963/crack_detection_and_measurement_utilizing_image_based_reconstruction.pdf?sequence=1

Zhou, S., Chen, Y., Zhang, D., Xie, J., & Zhou, Y. (2016). CLASSIFICATION OF SURFACE DEFECTS ON STEEL SHEET USING CONVOLUTIONAL NEURAL NETWORKS, 2016–1. https://doi.org/10.17222/mit.2015.335