Vision-Based Road Construction Site Detection and Localization

Krishna Kumar Karthikeyan

School of Electrical Engineering

Thesis submitted for examination for the degree of Master of Science in Technology.

Espoo 23.11.2020

Supervisor

Prof. Yu Xiao

Advisor

MSc. Aziza Zhanabatyrova

Aalto University
School of Electrical Engineering

Aalto University
School of Electrical
Engineering

Aalto University, P.O. BOX 11000, 00076 AALTO
www.aalto.fi
Abstract of the master's thesis

Author Krishna Kumar Karthikeyan

Title Vision-Based Road Construction Site Detection and Localization

Degree programme Automation and Electrical Engineering

Major Control, Robotics and Autonomous Systems **Code of major** ELEC3025

Supervisor Prof. Yu Xiao

Advisor MSc. Aziza Zhanabatyrova

Date 23.11.2020 **Number of pages** 51 **Language** English

Abstract

Autonomous driving is rapidly improving as companies and researchers are racing to create the perfect system. One of the uncertain environments which these systems face is when there is a construction site on the road. Road construction sites are dynamic and the components of the construction site can vary drastically from one site to another.

The goal of this thesis work was to explore approaches for vision-based detection of construction sites on roads using images taken from a single camera such as a dashboard camera. This would be helpful for autonomous driving and navigation solutions once the construction work has been identified and localized for making informed decisions.

Custom datasets have been created using existing NuScenes dataset as a starting point. Images containing road construction work from three cities - Boston, Helsinki and Singapore are included in the datasets. The two created datasets are targeted for image classification and object detection problems. Deep convolutional neural networks (CNN) based algorithms were tested on the datasets to classify and detect road construction sites. The final results of this work provides a proof of concept for detecting and localizing construction sites using a vision based system with support for future expansion.

Keywords Road Construction Site Detection, Image Classification, Object Detection, Image Dataset

Preface

I want to thank Professor Yu Xiao and my thesis advisor Aziza Zhanabatyrova for their guidance and support through my entire thesis.

This research was conducted as a part of the Mobile Cloud Computing research group under Communication and Networking Department at Aalto University. I would like to thank the research group and its members for providing insights and guidance during the course of my thesis.

Finally, I would also like to thank my friends and family for providing support during my studies.

Otaniemi, 23.11.2020

Krishna Kumar Karthikeyan

Contents

Abstract **3**

Preface **4**

Contents **5**

Abbreviations **7**

1 Introduction **8**
1.1 Thesis scope and objective . . . 8
1.2 Research questions . . . 9
1.3 Overview of thesis structure . . . 9

2 Background and Related Work **10**
2.1 Computer vision problems . . . 10
2.1.1 Classification problem . . . 10
2.1.2 Object detection problem . . . 11
2.1.3 Segmentation problem . . . 12
2.2 Evaluation metrics . . . 14
2.2.1 Metrics for classification . . . 14
2.2.2 Metrics for object detection . . . 16
2.3 Machine learning . . . 18
2.3.1 Supervised learning . . . 18
2.3.2 Unsupervised and semi-supervised learning . . . 19
2.3.3 Transfer learning . . . 20
2.4 Construction work on roads . . . 21

3 Experiments and implementation **26**
3.1 Dataset analysis . . . 26
3.1.1 KITTI dataset . . . 26
3.1.2 Cityscapes . . . 26
3.1.3 Mapillary Vistas . . . 28
3.1.4 BDD100K . . . 29
3.1.5 NuScenes . . . 30
3.2 Custom dataset . . . 31
3.3 Classification . . . 32
3.3.1 Resnet . . . 34
3.3.2 Densenet . . . 37
3.4 Object detection . . . 39
3.4.1 Labelling tool . . . 41
3.4.2 Method 1 - dataset for detection . . . 42
3.4.3 Yolo V3 . . . 43
3.4.4 Results . . . 44
3.4.5 Method - 2 Using the NuScenes dataset . . . 46

3.4.6 Retinanet 46
3.4.7 Results 46

4 Conclusion and Future Work **48**

5 References **49**

Abbreviations

AP	Average Precision
CNN	Convolution Neural Network
CVAT	Computer Vision Annotation Tool
FN	False Negative
FP	False Positive
GPU	Graphics Processing Units
GPS	Global Positioning System
IMU	Inertial Measurement Unit
IoU	Intersection Over Union
JSON	JavaScript Object Notation
mAP	Mean Average Precision
TN	True Negative
TP	True Positive
YOLO	You Only Look Once

1 Introduction

In recent years, autonomous driving has improved rapidly as companies have solved some of the major hurdles towards bringing out reliable and completely autonomous driving solutions. One of the important situations where the reliability of the system is tested is under unknown or changing circumstances. An example of a situation like this would be a construction work happening on a road. There is a large amount of randomness and variation in how the construction site looks. The autonomous system should be robust enough to react to such situations. The primary hindrance to such a system would be to identify and localize construction work on the map so that other autonomous cars can get updated information regarding their surroundings and approach such zones with increased caution.

The trivial approach to handle unknown risky circumstances is to avoid the situation altogether. This would be the first idea that comes to mind while thinking about navigating through construction sites on roads autonomously. Until the technology matures sufficiently to provide a robust solution which is capable of navigating through challenging environments such as road construction sites, the easier solution would be to avoid such regions. The lesser the risk, the safer it is for an autonomous system to make logical decisions.

The human element of uncertainty is another reason for drastic visual variations in what constitutes a construction site. Two construction sites working on similar issues might appear extremely different from each other even though they are within the same city and are operated by a single maintenance company. For example, at one of the construction sites, the workers might have forgotten to place a construction sign in the vicinity of the construction site even though the rules and regulations try to provide guidelines for the same.

1.1 Thesis scope and objective

Machine learning-based computer vision algorithms have become more robust and offer higher performance compared to traditional computer vision algorithms for vision-based tasks. The dataset selected plays a key role in training the machine learning models to perform specific tasks. This thesis would focus on discovering some of these relevant datasets and critically analyzing their suitability for the task of road construction work detection. An extension to existing datasets is also proposed for making the dataset more diverse and problem-specific.

The goal of this thesis would be to develop a vision-based system that can detect construction work happening on the roads so that they can be localized on a global map based on GPS coordinates. The algorithm would be using images captured with a single monocular camera such as a common dashboard camera present on a car and not any special stereo camera or infrared camera. Existing state of the art object detection algorithms are tested and tuned to verify their applicability for this specific problem.

The GPS coordinates would be obtained from a GPS unit in the car where the images or video is captured. This information regarding construction work

could further be utilized by autonomous vehicles to either alert the driver about the upcoming area where construction work is happening or could request for supervision from a remote driver. The system could also reroute the vehicle to avoid the construction zones and reach the required destination in the same time.

This work is limited to the research findings and goals achieved in the six months allocated for completing this master's thesis work. Experiments were carried out scientifically and the findings are reported in the results section. The thesis work was carried out keeping the long term goals and the overall project in mind. Due to the lack of any dataset for this specific task, this work focuses on creating an image dataset for this task keeping future expansion in mind.

1.2 Research questions

Construction work detection and localization is only a small part of the larger goal of autonomous driving. Scientifically solving a problem involves identification of the obstacles and tackling and solving smaller research problems. The main research question this thesis tries to address is to create a foundation for a system capable of detecting and localizing construction sites on roads using vision-based techniques keeping crowdsourcing in mind.

1.3 Overview of thesis structure

This thesis work is divided into four parts. Chapter 2 highlights some of the fundamental concepts and background information. This chapter also discusses the motivation behind creating a system to address this problem. Previous work related to construction work detection is also discussed. Chapter 3 discusses the experiments and work done in this thesis work. This is followed by concluding thoughts and possible future work related to this thesis in Section 4.

2 Background and Related Work

The coloured images from a camera are inherently ideal for distinguishing edges of objects based on complementary colours. When the foreground and background are composed of colours which are not very distinguishable, or when lighting conditions are poor, the camera would not suffice for identifying edges of different objects in the field of view. The camera input is also a very good source for identifying different objects on the scene as object detection and classification have matured significantly compared to the past due to easy access to large image datasets and advances in computer vision and machine learning. One of the major pitfalls of image-based detection, classification and localization is the inability to perceive depth. Image-based approaches for perceiving depth have been a point of interest for researchers from a very long time [1]. Other sensors such as lidars and radars are sensors with a very good capacity to perceive depth. Nevertheless, these are expensive sensors and not commonly available in the consumer market as a camera. Cameras have become a common device which almost all people have access to. Nowadays, many cars come equipped with a dashboard camera system which is typically used for recording and playback functionality. These cameras serve as an invaluable resource for capturing street-level images. This is a great source of information for creating datasets and real-time crowdsourced information for autonomous driving.

In this section, we look at some of the basic computer vision problems which can be solved from images, the tools needed for solving these problems such as machine learning and some basic concepts of machine learning which will prove invaluable when working on these problems.

Construction work on roads is discussed highlighting the motivation behind this work. A summary of related research work addressing similar problems is also provided.

2.1 Computer vision problems

Computer vision problems can be broadly classified into different categories based on the type of problem being solved and the type of output which we expect from the algorithm. Some of the relevant types of problems are discussed in the following subsections.

2.1.1 Classification problem

Classification problems involve identifying which category a particular input belongs to. Based on an input vector x, a corresponding valid output category $y \subset \{1, 2, \ldots, k\}$ is selected by a statistical algorithm. Machine learning involves framing a model $f(x) : \mathbb{R}^n \rightarrow \{1, 2, \ldots, k\}$. The generated model $f(x)$ takes the input vector x and outputs an integer y which corresponds to the category that the input belongs to [2]. If needed, the model can be created such that it outputs a number between 0 and 1 which represents the probability that a given image belongs to the predicted class. This probability is a useful metric which signifies how good the model believes the prediction to be.

In the case of image classification, the input will be an image and the algorithm would decide what category the image belongs to. Usually, the image would contain an object of interest and the algorithm will be selecting which category the object belongs to. One of the early examples of image classification which is referred to even today is the MNIST dataset [3]. This problem involves single digit recognition where images of handwritten digits are captured and the goal is to create an image recognition algorithm which can take an image of a single-digit and categorize it into one of ten categories corresponding to the digits from 0 to 9. An overview of the process is shown in Figure 1. In more complex scenarios, the dataset might contain larger images of objects which need to be classified or more number of classes which the classifier needs to pick from. The imagenet dataset [4] is an object detection image dataset which was released in 2009. The dataset contains over 14 million images which belong to 21841 categories [5]. The dataset tries to provide on an average 1000 images for each category.

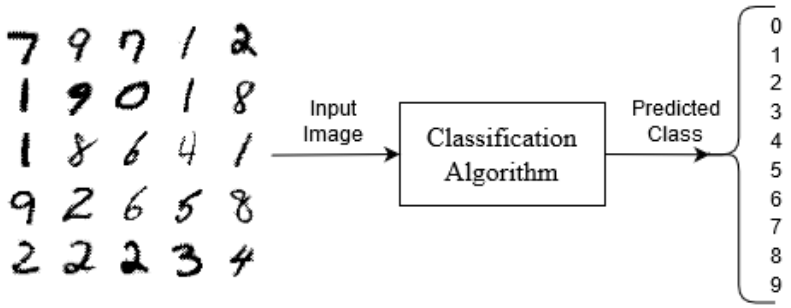

Figure 1: Example of a simple image classifier with input from the MNIST dataset.

Generally, the categories are referred to as classes in the machine learning domain. Based on the number of classes being classified, the problem can be either a binary classification problem where there are only two classes to be categorized or a multi-class classification problem where the number of classes is more than two.

2.1.2 Object detection problem

The previous section highlighted a type of problem where the goal was to identify the type of object present in an image. The whole image would correspond to a single class and the algorithm is supposed to select a class from a list of probable classes. Real-world scenarios demand a lot more than this as a typical scene or image would contain much more irrelevant information in the background.The algorithm is expected to differentiate between the object and the background while simultaneously detecting which object is present in the image. This type of problem is typically referred to as object detection and localization problem.

A famous example image for object detection is shown in Figure 2. The algorithm can detect three objects in the image, a dog, a bicycle and a truck. The algorithm is robust enough to distinguish objects from the background and create bounding boxes around the objects of interest. The second part of the problem involves identifying the object inside the bounding box and assigning a category to it.

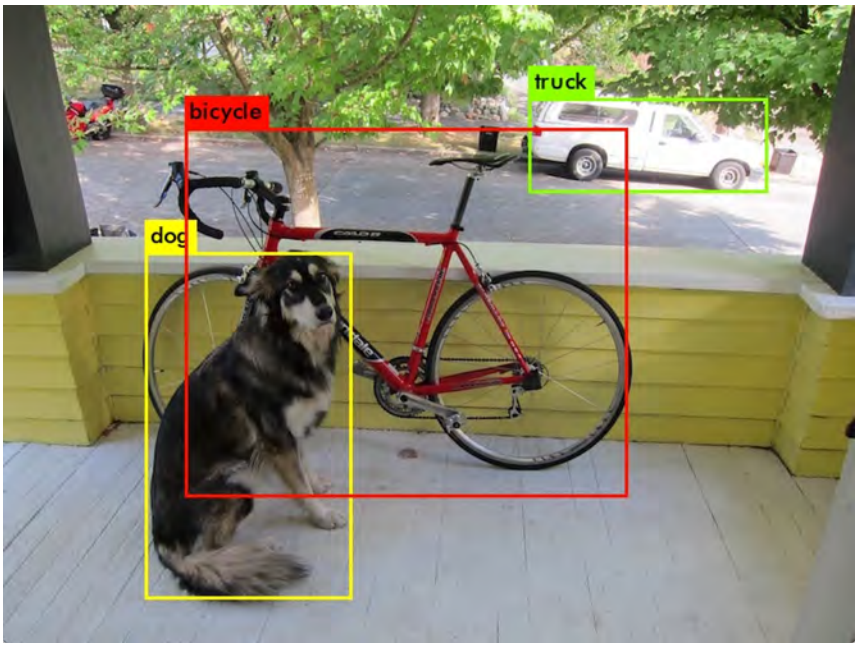

Figure 2: Sample output of an object detection algorithm.

This image is a good example as it also highlights some of the common problems faced by image-based object detection. The first problem is occlusion, the objects in the front block the objects behind either partially or completely. The dog in the image is partially covering the bicycle. The second problem is the variation in lighting conditions. The truck in the image is very brightly lit and overexposed compared to the dog and the bicycle. The algorithm should be designed keeping these variations and limitations of image-based detection methods.

2.1.3 Segmentation problem

Object detection problems involve identifying bounding boxes around objects of interest and identifying what object is present in the bounding box. This is still a bit vague as the bounding box typically would include objects in the background as well. In Figure 2, almost one-third of the bounding box for the bicycle is covered by the bounding box for the dog. Semantic segmentation tries to address this problem.

Semantic segmentation involves assigning a class for every pixel in the image. The goal of the algorithm is to identify what object each pixel belongs to with a corresponding confidence score. Figure 3 shows an image where the classes have been encoded with colours and drawn over the image as a transparent layer. For example, the green parts indicate vegetation or trees, the sky is highlighted in blue, cars are a darker shade of blue and so on.

This kind of object identification at a pixel level accuracy is much better than the bounding box detection for identifying objects. Semantic segmentation tries to understand the whole scene presented to the algorithm as an image. The drawback this algorithm faces is when objects belonging to the same class are present as a cluster within the image. The algorithm outputs just a large blob with the same class

Figure 3: A visualization of semantic segmentation from the Cityscapes dataset.

and will not be able to distinguish individual elements. For example, in Figure 3, The bicycles on the right of the image look like a continuous blob of pixels belonging to the bicycle class although they are two separate bicycles. This problem is addressed by another type of segmentation problem called as instance segmentation.

Instance segmentation also involves pixel-wise labelling of points. The difference from semantic segmentation is that individual instances of the classes are identified separately. An example of instance segmentation is shown in Figure 4 where each instance of a car is highlighted by a different colour even though some of them are adjacent to one another.

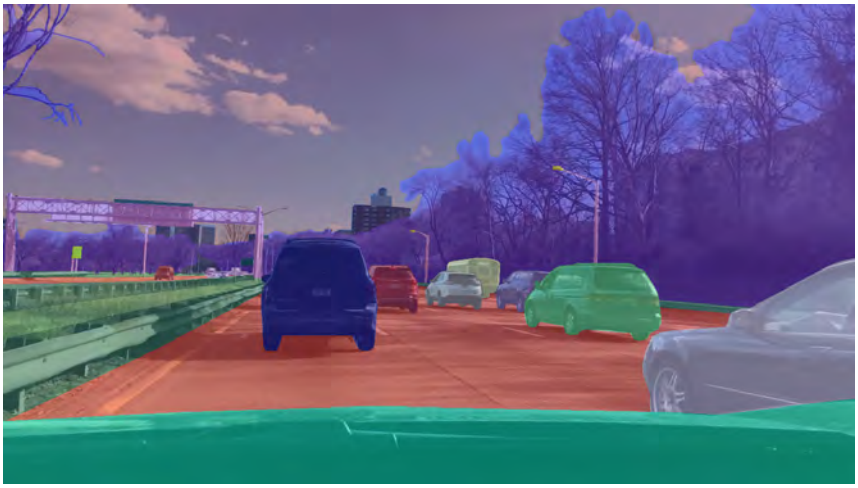

Figure 4: A visualization of instance segmentation from the BDD100K dataset.

Image classification was the simplest problem in terms of complexity amongst the computer vision problems discussed. Object detection and localization is a problem which has higher complexity. Semantic segmentation and instance segmentation follow and have increasing complexity. The hierarchy of complexity of these computer vision problems is shown in Figure 5. Complexity being referred to here is the computational complexity and the effort needed to label the data and train the model.

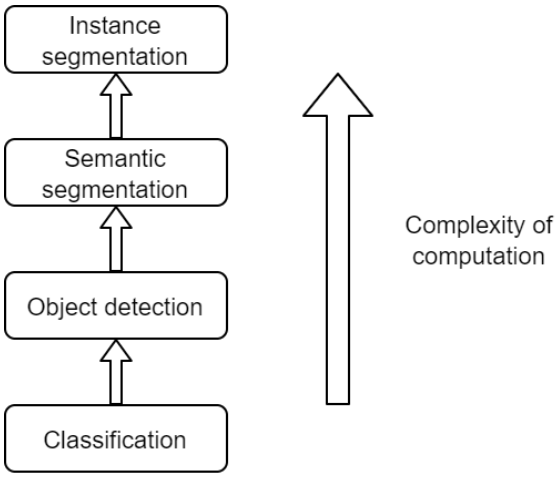

Figure 5: Hierarchy of complexity of computer vision problems.

The following section will discuss some of the relevant evaluation metrics for analyzing the performance of models trying to solve these computer vision problems.

2.2 Evaluation metrics

Evaluation of performance in a quantitative fashion is an important factor while designing methods to solve the computer vision problems discussed in section 2.1. The type of evaluation metric used is selected based on the type of problem being addressed. Image classification would require different metrics compared to object detection.

2.2.1 Metrics for classification

Accuracy is the common metric which comes to mind when we begin evaluating a models performance. It is a quantitative measure to judge how good the designed system performs. Mathematically, accuracy can be represented as in equation 1.

$$Accuracy = \frac{\text{correct predictions}}{\text{total number of samples}} \tag{1}$$

Although this metric seems intuitive and good for performance evaluation, the quantitative value is misleading at times. For example, if we consider an example of whether a car would meet with an accident or not, the occurrence of the accident is much rarer compared to the latter. Hence, if the model predicted ten out of ten times that there would be no accident and only one instance has an accident, the accuracy of our model would be 90%. In this scenario, predicting the positive scenario of the accident is much more critical than the absence of the same. The 90% accuracy metric is misleading in this case.

Table 1 is a method to visualize the actual findings while evaluating a model. During classification, the number of predictions which were classified correctly as a given class is indicated as true positives (**TP**). The absence of a particular class

predicted correctly corresponds to true negatives(**TN**). Similarly, false positives (**FP**) occur when the model classifies the input as a particular class although the class is not present in the input sample. False negatives (**FN**) occur when the input contains the particular class and the model incorrectly predicts the absence of the class for the input.

		Actual Class	
		Positive	Negative
Predicted Class	Positive	**True Positive (TP)**	False Positive (FP)
	Negative	False Negative (FN)	**True Negative (TN)**

Table 1: Confusion matrix for binary classification.

Several other metrics can be derived by using the numbers generated in the confusion matrix shown in Table 1. Accuracy can be mathematically defined as described in 2.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{2}$$

Accuracy is a good and valid metric to be used in classification problems if the dataset has a proportional distribution of all classes and has no significant class imbalance. If the class to be identified is very sparse and the system is expected to have least amount of false positives for the sparse class, accuracy is not a good metric.

Precision is another metric used for evaluating the performance of the model. Precision corresponds to the ratio of the true positives to the total predicted positives by the model. This metric is used to address false positives. If a model has higher precision, it means that the model will make lesser number of false positives. Mathematically, precision can be described as in equation 3.

$$Precision = \frac{TP}{TP + FP} \tag{3}$$

Precision is the metric to be focused on when dealing with critical systems which cannot have false positives or need to minimize the number of false positives.

Recall is the metric which denotes what proportion of the actual positives are correctly classified (equation 4).

$$Recall = \frac{TP}{TP + FN} \tag{4}$$

Recall tries to emphasize on making positive predictions. As all the metrics discussed above has some positive aspects and some shortcomings, more complex metrics are needed to quantitatively evaluate the performance of a model.

The F1 score is a complex derived metric which varies between 0 and 1. It is the harmonic mean of precision and recall as described in equation 5.

$$F_1 = 2 * \frac{precision * recall}{precision + recall} \tag{5}$$

Training a model to improve on its F1 score tries to create a solution which has both good precision and recall. In a practical scenario, this seems like a good solution which overcomes some of the shortcomings of using only precision or recall independently. This is not particularly true as the F1 score gives equal importance to both precision and recall whereas in certain problems, they might need different weightings of precision and recall while evaluating the performance of the model.

Crossentropy loss or log loss is another evaluation metric for binary classifiers. The formula for the same is described in equation 6 where y is the label - 0 for one class and 1 for the other class, $p(y)$ is the predicted probability of the input belonging to the particular class.

$$\text{Binanry crossentropy} = -\frac{1}{N}\sum_{i=1}^{N} y_i \cdot \log\left(p(y_i)\right) + (1 - y_i) \cdot log(1 - p(y_i)) \quad (6)$$

Log loss takes into account the probabilities of the predictions made by the model and gives a more accurate view of the performance of a binary classifier. A slight problem arises when the dataset used is an imbalanced dataset. The class imbalance can be addressed by using different techniques as discussed in a later section.

Log loss is typically used as the objective function while training machine learning based classifiers and other evaluation metrics are used for evaluating model performance.

2.2.2 Metrics for object detection

Object detection is a two-fold computer vision problem which involves localization of an object and classification of the category of the object. Although evaluating the performance of the object detection model is not as simple as the classification problem, it follows similar patterns when defining metrics.

Intersection Over Union (IoU) is a common term when talking about object detection. It is computed as a ratio of two areas, the intersection area between the true ground truth bounding box and the predicted bounding box to the total union area of both the bounding boxes. A visualization of computing IoU is shown in Figure 6. A perfect prediction implies that the predicted bounding box is overlapping completely with the ground truth bounding box and the IoU, in this case, will be 1. IoU is a number between 0 and 1 and a higher IoU indicates a better prediction of the bounding box.

As perfect detection of objects are hard and we need to create an acceptable range of error during object detection, we will have to set a nominal value for the IoU before computing the overall performance of the object detection algorithm. For example, if we set an IoU threshold of 0.5, it means that we consider a prediction to be correct if the predicted bounding box overlaps at least 50% with the ground truth bounding box. This is how a True Positive is defined in the case of object detection. Similarly, if the predicted bounding box has an IoU lesser than the selected threshold of 0.5, the prediction is deemed to be a False Positive. If a ground truth box is present in an image and the model fails to predict any bounding boxes for that image, it is considered a False Negative. True negatives are not considered a good metric

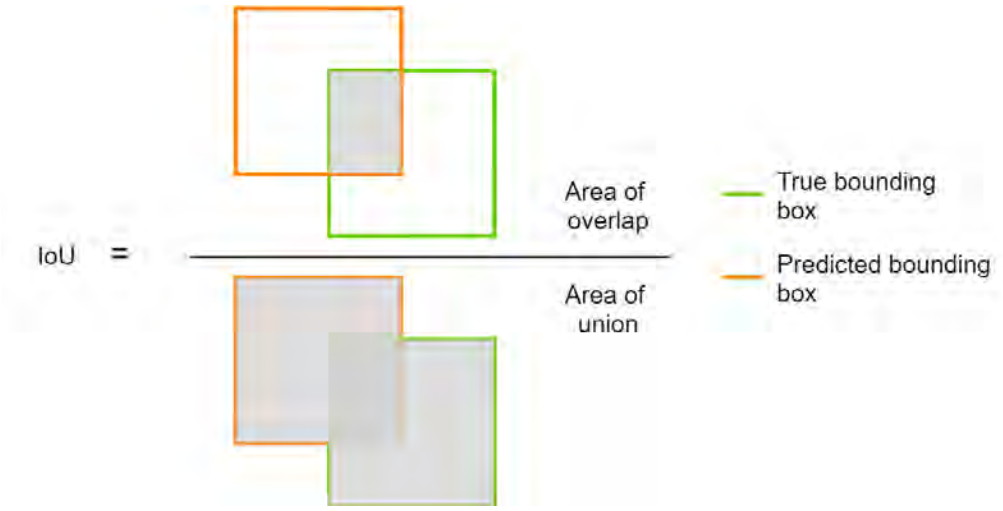

Figure 6: Intersection over union (IoU) for object detection.

for object detection and is not used as it corresponds to all parts of the image where we didn't predict any bounding box and there is no ground truth box.

The threshold for IoU is a key factor when evaluating models. If a very precise bounding box is needed, a higher value can be selected such as 0.75. The selection of this parameter varies based on the critical nature of the problem and the complexity of the dataset.

Precision and recall metrics for object detection are computed in a similar fashion as classification. Precision can be calculated using equation 3 and recall using equation 4. As TP, TN and FP are defined based on the threshold for IoU, we obtain different values for precision and recall by varying the threshold value. If we plot precision against the recall values in a graph while varying the threshold value, we obtain the precision-recall graph from which we can derive other metrics such as **Average Precision (AP)** and **mean Average Precision (mAP)**. An example precision-recall plot is shown in Figure 7. The blue line indicates the actual precision values obtained and the orange line is the estimated precision value corresponding to the recall values in the X axis. As this curve is always a monotonically decreasing curve, the highest value of precision to the right of the current recall value is selected and the plot for the estimated precision is obtained.

The area under the precision-recall curve corresponds to the Average Precision (AP) for object detection. Some object detection competitions such as the PASCAL VOC [6] and MS COCO [7] use a metric called as mean Average Precision or mAP for evaluating performance. The evaluation of this metric varies slightly depending on the competition. The basic principle remains the same, which is to compute the interpolated average precision. This is done by sampling the precision values at regular intervals along the curve and determining the average value of precision. When this process is repeated for every class in the dataset, we obtain the AP values of the corresponding classes. The mAP value is evaluated by computing the average of the AP values for all the classes. The difference between the PASCAL VOC mAP and the MS COCO mAP is that VOC uses a 11-point interpolated average and

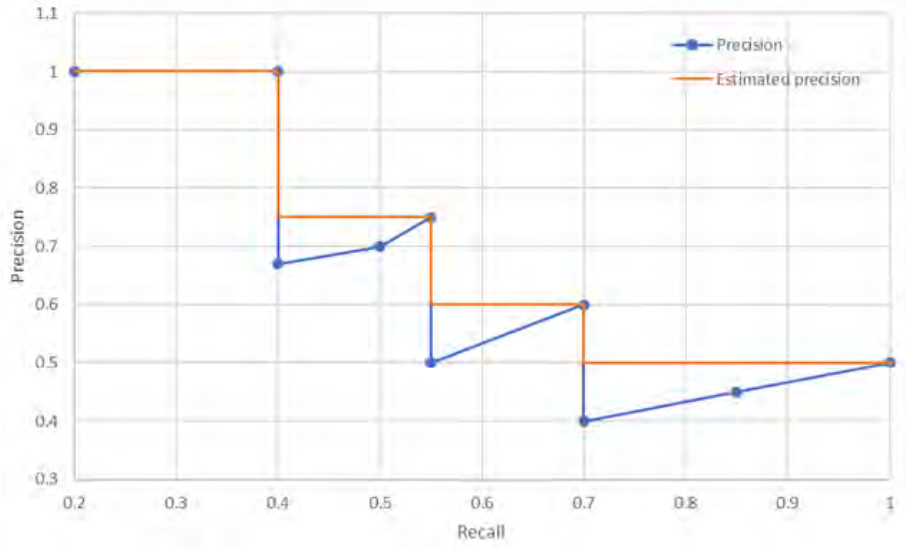

Figure 7: Example precision-recall plot.

COCO uses a 101-point interpolated average for computing mAP.

The following sections will discuss the tools needed for solving these computer vision problems. As these problems have varying levels of complexity and involve large number of mathematical computations, we will have to leverage tools such as machine learning and other techniques within the domain for solving the problem at hand.

2.3 Machine learning

Machine learning relates to the study of statistical computer algorithms used to identify logic and patterns in data. Due to the surge in computational power over the last decade or so, researchers have been able to leverage larger and more complex datasets to find logical insights.

Machine learning algorithms try to understand the underlying pattern in the data and this can be a crucial input for making predictions in the future for the same problem. These algorithms have already made their way into our daily lives and have become an integral part our lives.

Instead of delving into the nuances of neural networks and how machine learning works, solving the current problem demands a certain level of abstraction and higher-level approach. The basic principles and working of neural networks would not be discussed in detail. The overall working of some of the algorithms used and certain aspects related to their working will be discussed in further sections. This thesis tries to solve a practical problem with the tools of deep learning and computer vision.

2.3.1 Supervised learning

Supervised learning is a type of machine learning algorithm which learns patterns in historic data which has already been labelled. This basically means that the entire

data which is given as input to the algorithm needs to contain labels for the specific type of problem which is expected to be solved. The algorithm trains itself to create a function which maps an input vector to an output vector based on the example input-output pairs in the labelled training data.

The key for a supervised learning algorithm to work is the dataset. The algorithm learns to generalize the structure and patterns found in the input-output pairs to create an inferred function. The dataset should be diverse and representative enough to cover expected behaviour to unknown inputs.

Models which are trained to find these structural relationships in the data are associated with a complexity similar to the complexity of solving polynomial functions with increasing order. If the dataset is small and does not generalize all scenarios, using a complex model could lead to overfitting where the machine learning model knows the training data too well and follows the errors and noise in the dataset too closely. In other words, the model would not learn the trends in the data and memorize the training input and output relationships. Hence, an optimal model complexity needs to be selected for a particular problem and dataset.

2.3.2 Unsupervised and semi-supervised learning

Unsupervised learning involves creating algorithms which work on datasets which do not consist of labelled data. Creating or capturing unlabelled data is a much simpler task compared to labelling the data. Labelling takes up a significant part of the work compared to other tasks while creating a dataset. This time is much more significant when the dataset being created is related to images or video. Depending on the complexity of the problem (as shown in Figure 5), the time for labelling also increases. Labelling for datasets targeting to solve more complex problems such as semantic segmentation is prone to subjective errors even when a lot of time is dedicated for labelling.

Unsupervised learning tries to identify patterns and structure in the data which could be difficult to find when done manually. The features identified by unsupervised learning algorithms could be useful for manual categorization. It could be a useful tool for working with large datasets.

Unsupervised learning is used for problems where the data needs to be clustered into groups or other problems such as dimensionality reduction. K-means clustering is an example of an unsupervised learning algorithm which tries to group the data into k different clusters. An example is shown in Figure 8 where the data has been grouped into three different clusters indicated by the different colours.

Semi-supervised learning is another approach to training a model on a dataset where a small part of the dataset consists of labelled data and a larger part of unlabelled data. This type of learning algorithm falls somewhere between supervised and unsupervised learning algorithms. When a certain amount of labelled data is used with the unlabelled data, there will be a considerable improvement in the learning accuracy and training time compared to purely unsupervised learning.

Supervised learning is the typical method used for solving machine learning problems related to classification, regression, object detection and other similar

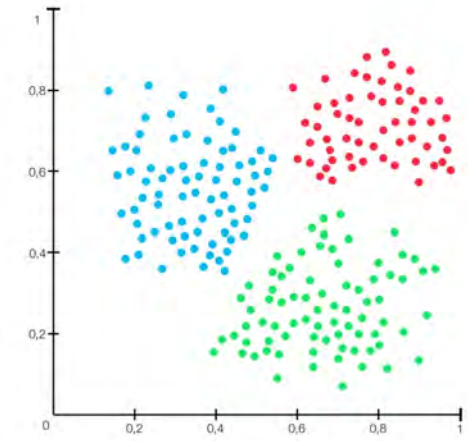

Figure 8: A visualization of k-means clustering problem. k is set to 3 to identify 3 clusters.

applications. Classification is a problem where the model outputs a discrete value corresponding to different classes. In regression, the model outputs a real numbered value from a continuous output space. The model tries to fix a continuous-valued function to the training data. A sample visualization of classification and regression problems is shown in Figure 9.

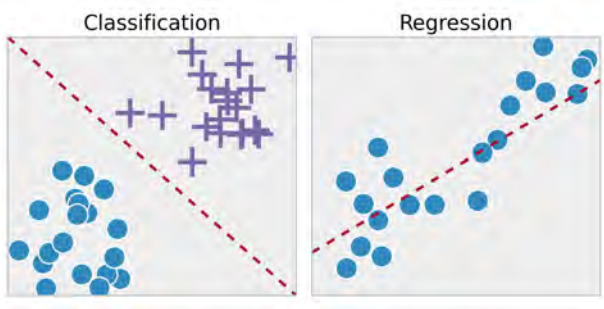

Figure 9: A visualization of classification and regression problems.

As the current problem requires object detection and scene understanding, the type of machine learning algorithms will be supervised learning. As stated earlier, the dataset plays a key role for the supervised learning algorithm to perform well and the dataset should be diverse and representative of the scenarios where the algorithm is planned to be used.

2.3.3 Transfer learning

Transfer learning is a machine learning concept which originated from the days of statistical modelling taking inspiration from how humans learn. When a human has to learn a new concept to solve a new problem, he tries to relate to the concepts and fundamentals which he learnt while solving other related problems and learns

to solve the new problem. A similar approach can be used when we try to solve machine learning or deep learning problems. Algorithms try to leverage previously learnt knowledge when solving new problems.

During transfer learning, we can use parts of the model which is trained on labelled data from one domain to solve a problem in another related domain [8]. This idea would help the algorithm to learn faster and create a model which can generalize well for the new problem.

Image classifiers and deep neural networks which are used for object detection are complex computational algorithms and need significant time for performing backtracking and update operations through all the layers of the network. Some of this computational time can be reduced by using hardware accelerated Graphics Processing Units (GPU). As datasets get larger and image resolutions become higher, computations take a longer time and the overall training of the model takes a very long time even when trained with a GPU. This is where transfer learning reduces the training time. Depending on the target level of accuracy to be achieved, resolution of the images and the similarity of the problems, the reduction in the training time will be affected.

If we use transfer learning in the domain of computer vision, we can obtain models which are pre-trained on large image datasets such as the Imagenet challenge [4] and use them as a starting point for solving our custom computer vision problem. In simpler terms, if done right, the model will remember to identify certain low level features such as shapes, edges, corners and light intensity from the domain it was trained on [9]. While training the same model for another task related to computer vision, the "knowledge" the model gained from the first task would help in reducing the training time and reach a better accuracy sooner for the second task.

For addressing the problem of construction work detection, transfer learning will be employed as one of the policies during the training process for getting a better starting point and to make the training process faster. The specifics of how the transfer learning is done in each algorithm will be highlighted in the later sections where the implementation and experiments are discussed.

2.4 Construction work on roads

Construction work and maintenance work on roads has become a common sight on modern roads in both developed and developing countries. These kinds of situations are almost always associated with a risk factor caused by several dynamic unknown factors. Systems in autonomous vehicles should be robust enough to navigate the modern congested road network safely. This section discusses relevant work carried out in the research field and the motivation for this thesis work.

Wimmer et al. [10] discuss a method for generating a highly accurate "Road work map" using sensors including a video camera and a laser scanner. The system was designed keeping driver assistance systems in mind. They felt that the digital representation of the road would help drivers make assisted decisions regarding their planned navigation. The main drawback of this approach is that the authors use laser scanner and video data to distinguish "beacons" or delineators on the road used

to mark a construction site on the lanes. This might not always be the case and the current goal is to create a purely vision based system.

The authors of [11] try to model a probabilistic approach to construction site detection on motorways. The authors design a Bayesian network which takes certain parameters as input and predict the possibility of occurrence of a construction site in the road ahead on the motorway based on observed parameters such as lane markers, construction site separators, speed limit changes, construction road signs and many more. Although this method provides a probabilistic approach to estimating construction sites, it is not a definitive way of identifying and distinguishing construction sites on the road.

Detection of hydraulic excavators and dump trucks is a narrowed down and specific problem which is addressed by Wenyang Ji et al. [12]. This work tries to assess methods for detecting hydraulic excavators and dump trucks from video and image sources. An interesting aspect of this research work is being able to detect some of the vehicles associated with road construction work and other constructions too. The method proposed by the authors involves traditional computer vision techniques such as edge detection and generalizing geometric patterns associated with these vehicles after applying pre-processing steps on the image. This method seems a bit crude and is not a good fit for detecting construction works on roads in general.

The work by Felix Bröker [13] is the closest relevant work in the academic field to the task at hand. The work focuses on identifying construction sites on motorways using a camera based approach. The goal of this work is to aid in autonomous driving directly by creating a pixel-level accurate representation of the construction site on the road. The author also uses a systematic approach of addressing the problem as a classification problem before targeting segmentation. For the classification task, the author tries to distinguish between "positive" images containing construction work and "negative" or "background" class of images. The work also addresses the lack of datasets focused on construction work on roads and a necessity for creating one. A dataset with around 800 images for each class was created followed by augmentation for improving the number of samples. The work also used a self-created 311 image dataset with segmentation annotation for distinguishing drivable road from construction work and background. The resolution for the chosen images was 128x128.

Some differences between the methods discussed above and this work is that the current work is more focused on building an application oriented solution around the problem keeping in mind the computational constraints and resource limitations allotted for this task. The goal would be to provide a framework which can be expanded to more geographical locations if needed and take the crowd-sourcing aspect into account.

Construction sites are dynamic and could evolve at different paces. Some examples are as follows. If there is a pothole on the road and it has to be plugged, if the pothole is small, the construction work on the road might not disrupt the entire traffic flow on that road. The lane in which the pothole needs to be fixed might be blocked for a short period of a few hours at most (Figure 10). On the other hand, if there is a gas line leak which needs to be fixed, the whole section of the road might

have to be excavated and rebuilt once the repair work gets completed. In this case, the entire road might be closed for a few weeks. An example of a road closed for construction work is shown in Figure 11. Depending on the nature of the work, the time needed and the extent of the closure of the road varies.

Figure 10: Lane blocked for fixing a pothole.

Figure 11: Road closed for construction work.

Construction sites on the road are usually characterized by some common factors no matter which country the site is in. There would be variations in standards of colours, placement of signs and shapes of other objects such as delineators or cones. Most construction sites are required to have a street sign indicating construction work in the vicinity of the construction sites. Figure 12 is an example of a construction site captured in Singapore where the construction sign is absent and only a caution sign for merging lanes is placed along the approach to the construction site. In a logical sense, this sign is appropriate as it also indicates to the human driver as to

where the construction work or obstacle in the road is, and which lane the driver needs to take.

Figure 12: Construction site in Singapore where construction sign is not very clear.

Figure 13 shows another example of a temporary construction work where only a temporary caution sign is held up to alert drivers about the construction work. This is another variation of a construction site scenario where the driver needs to exercise caution when approaching the zone and autonomous systems need to be robust to know what to do.

Figure 13: Construction site Finland with a temporary caution sign.

Figure 14 shows some other variations in construction sites from Boston, Singapore and Helsinki. Looking at these images and the drastic differences in the signs and standards, it can be concluded that using only these signs is not a very reliable way of identifying a construction site. There are also instances when the construction sign

might not be placed at an appropriate distance from the construction site. Hence, an alternative approach involving identifying a construction site using aspects other than a construction sign is needed.

Figure 14: Variations in construction signs captured in USA, Finland and Singapore.

Although there are existing databases for the road infrastructure in each country independently maintained by local authorities and publicly available detailed maps such as google maps, here maps, and other mapping services, there is a lack of a central live database of road construction work happening in a country [14]. This could be due to many reasons, the main one being that there is no single contractor or construction company which would be responsible for the road construction and repair work for an entire country or state. This lack of communication and creation of a common live database adds to the randomness of the whole process. It is difficult and complicated to build a single system which is capable of tracking all the construction sites being carried out by different companies in different parts of the country. Hence, to bring a level of automation to this process and to make it scalable in different geographical locations, a crowd-sourced approach will be beneficial. The following section discusses the existing datasets and the necessity for creating a custom dataset for solving this problem.

3 Experiments and implementation

Similar to any other deep learning based problem, to solve the problem of construction work detection, we need to start with identifying or creating a relevant dataset and tackle the problem in a systematic way.

3.1 Dataset analysis

In the last decade, there have been numerous image and video datasets targeted towards getting better insights into autonomous driving. None of these relevant datasets focuses on road construction work in particular due to many reasons. The primary reason being that road construction work occurs with much less frequency when we are driving on a road. These datasets usually focus on capturing more commonly occurring objects such as pedestrians, lane markings, traffic signs, and signals. As discussed earlier, the trivial approach to handle unknown risky circumstances such as construction sites is to avoid the situation altogether.

Capturing images or videos and creating a dataset from scratch focusing on road construction work would be a time consuming and expensive exercise. Existing datasets were leveraged to create a starting point for creating a new dataset for the problem at hand. A short analysis of some of the famous autonomous driving related datasets is discussed in the following sections.

3.1.1 KITTI dataset

The KIITI dataset was created by a team of researchers from the Karlsruhe Institute of Technology (KIT) in Germany[15]. The researchers created a vision benchmark suite for several tasks related to autonomous driving such as The dataset was captured using an array of sensors and cameras mounted on a car. A stereo camera setup with two monochrome and two colour cameras was used to capture video data. An on board GPS unit recorded the precise GPS location for each position of the car. The research team also used a laser scanner to provide ground truths for tasks involving 3D object detection and tracking. A normal station wagon car was equipped with all the sensors mentioned and driven around the German city of Karlsruhe in rural and highways. The images captured by the four cameras on the car have a very wide field of view and aspect ratio. The images have a resolution of 1392×512 as shown in Figure 15.

The KITTI dataset does not specifically focus on construction work on roads even with rare occurrences. The dataset is focused towards capturing real-world traffic situations. Also, the extra wide aspect ratio would create problems in the future as we have crowdsourced data in mind. Using the KITTI dataset as a starting point for the construction work detection dataset would not be feasible.

3.1.2 Cityscapes

The Cityscapes dataset is another famous autonomous driving related dataset created around 2016 [16]. The dataset was created keeping the complex urban street scenes in

Figure 15: Sample image from the KITTI dataset.

mind. The dataset is focused towards enabling semantic segmentation and instance segmentation. The dataset was created using frames extracted from a stereo camera setup mounted on a car. The data was captured during varying seasons including spring, summer and fall. Majority of the images were captured in 50 cities across Germany and a few neighbouring countries. The authors avoided capturing data during adverse weather conditions such as snowfall and rain as these would require additional sensors and more sophisticated techniques in order to create a reliable system. Unlike the KITTI dataset, the Cityscapes dataset a more common aspect ratio of 2:1. The images have a standard resolution of $2048X1024$ pixels. The training and validation components of the dataset are categorized into fine and coarse pixel-wise annotations. The test annotations consist of around 1525 fine annotated images.

The segmentation labels were classified into 30 predefined categories. These included both static and dynamic objects which we can see in a common driving scene. The categories are broadly classified into 8 different broad groups as highlighted in Table 2. A sample image with fine annotations for the instance segmentation problem from this dataset is shown in Figure 16.

Group	**Classes**
flat	road, sidewalk, parking, rail track
human	person, rider
vehicle	car, truck, bus, on rails, motorcycle, bicycle, caravan, trailer
construction	building, wall, fence, guard rail, bridge, tunnel
object	pole, pole group, traffic sign, traffic light
nature	vegetation, terrain
sky	sky
void	ground, dynamic, static

Table 2: List of classes in the Cityscapes dataset.

This dataset was analyzed as it is one of the famous standard datasets related to autonomous driving. The classes in this dataset focus on commonly occurring

Figure 16: Sample image from the Cityscapes dataset with fine instance segmentation.

objects while driving on the road such as vehicles and humans typically in ideal driving conditions. Hence, the cityscapes dataset was not selected as a baseline for creating a dataset for the construction work detection problem.

3.1.3 Mapillary Vistas

The Mapillary Vistas dataset [17] is another large-scale dataset which was partially open sourced in 2017. Unlike the previously discussed datasets, this dataset relies on data which is gathered by crowdsourcing. Their goal was to create a rich and diverse dataset which caters to the needs of autonomous intelligent driving by providing a set of images which are representative of real-world conditions. The widespread use of smartphones was a valuable resource which was the main input to the Mapillary platform [18], a platform for collaboration for creating a huge database of street-level imagery and map data. The images were handpicked from their massive collection of street-level images based on a series of stringent parameters. The creators of the dataset have assimilated images from numerous cities across continents. The dataset also includes images from varying weather conditions and times of day.

25,000 images were selected from the Mapillary platform and annotations were created for semantic segmentation and instance segmentation. The size of this dataset while considering fine-grained annotations, is around five times that of the Cityscapes dataset. A partial dataset with annotations for 66 classes of objects for semantic segmentation and 37 classes for instance segmentation was released by the Mapillary team for research and academic purposes. The complete dataset with annotations for 100 classes of objects for semantic segmentation and 58 classes for instance segmentation is available from Mapillary research with a commercial licence.

Although the Mapillary dataset provides a diverse and rich annotation based dataset, the annotated classes do not really focus on construction work on the road. The focus on crowdsourcing and diversity is great and in line with the goals of this thesis work. Unfortunately, the dataset is not a suitable candidate for this task of road construction work detection. The images were sampled at random from a large

set of street-level images and it was ensured that the multiple images were not selected from the same scene [17]. The task of filtering out images with construction work from the large dataset of thousands of images would involve unnecessary additional effort and time. Keeping all these factors in mind, the Mapillary Vistas dataset was not selected as a baseline for the current problem.

3.1.4 BDD100K

The Berkeley Deep Drive(BDD) dataset signifies another milestone towards autonomous driving. This famous dataset was created by the researchers at the Berkeley University and made openly available to researchers around the world during 2018 [19].

This is one of the more advanced and large scale datasets. The dataset focuses on addressing multiple problems related to autonomous driving. This open dataset consists of 100,000 high definition video sequences which were consolidated from 1,100 hours of driving data, hence the name, BDD100K dataset. The data was captured at different times of the day including night time footage. There are also variations in the weather in the captured footage.

Figure 17: Sample images from the BDD100k dataset with drivable lane segmentation, instance segmentation, road marking segmentation, and 2D object detection.

The dataset addresses the object detection problem related to autonomous driving. Bounding box annotations are provided for 10 categories of objects commonly found during driving. These include bus, traffic light, traffic sign, person, bike, truck, motor, car, train and rider. The annotations also include other parameters which indicate whether the annotated object is occuluded or truncated. This dataset does not focus on 3D object detection unlike the KITTI dataset. The bounding box annotations are for 2D objects. Hence the occlusion and truncation parameters are added to the data.

As highlighted earlier, this dataset does not limit to object detection. The authors also provide labels for instance segmentation, drivable area detection and lane marking detection. All these categories have pixel-wise labels. In addition to the categories selected for object detection problem, the authors provide semantic instance segmented labels for twelve more categories of objects. The drivable lane segmentation contains pixel-wise labels for the current lane the car is driving in and the alternate lanes. Only the drivable sections are labelled, avoiding all obstacles on the road. The lane marking segmentation includes categorized labels for the markings on the road including zebra crossings, white or yellow single or double lines. An example of each type of annotation in the dataset is shown in Figure 17.

The dataset was analyzed and the frequency of road construction work in the sequences of images was very low. Similar to the previously described datasets, the BDD100k dataset was also avoided for the current research work.

3.1.5 NuScenes

In March 2019, Aptiv Autonomous Mobility (formerly nuTonomy) released a massive dataset related to autonomous driving called the NuScenes dataset [20]. The engineering team has created a multi-modal dataset captured by a set of complementary sensors in cities from two different continents. The set of sensors includes six cameras, five radars, Inertial Measurement Unit (IMU), LIDAR, and a GPS unit. The sensors were mounted on a car and driven around the cities of Boston and Singapore. These cities were chosen because of their diverse natures of driving conditions and rules. Boston follows a right hand driving system whereas Singapore chooses a left handed driving system. Synchronization of the data captured from all these sensors was ensured by the research team. The images and lidar sweeps were synchronized and extracted at a fixed frequency. Hence, the images consist of continuous sequences of images covering all views from the ego vehicle.

As the dataset is targeted towards facilitating sensor fusion and making sense from different sensors simultaneously, it provides annotations for solving different problems. The dataset provides annotations for 3D object localization. Each object which is annotated has a class associated with it which is one of 23 different classes, 3D bounding box, and attributes for each frame they occur in. The 3D bounding boxes are cuboids saved with additional parameters such as visibility, activity and pose. The 23 classes are listed in Table 3. A total of more than 1.4 million bounding boxes have been annotated in the dataset.

Although this dataset is more focused towards providing a multimodal dataset with data captured from multiple sensors, the goal of the current task is oriented towards a crowdsourced image based approach for gathering data. Hence, only dashboard cameras and commonly available cameras were considered as input sources and not complex and expensive sensors such as radar or lidar. This limits the part of data needed from the large multi-sensor dataset.

The NuScenes dataset was selected as the most viable option which can be used as a base for creating a new dataset for construction work detection after careful analysis. The dataset contains annotations for a few relevant classes as seen

animal	**object - barrier**	**vehicle - construction**
pedestrian - adult	object - debris	vehicle - emergency ambulance
pedestrian - child	object - pushable-pullable	vehicle - emergency police
pedestrian - construction worker	**object - traffic cone**	vehicle - motorcycle
pedestrian - personal mobility	vehicle - bicycle	vehicle - trailer
pedestrian - police officer	vehicle - bus(bendy)	vehicle - truck
pedestrian - stroller	vehicle - bus(rigid)	object - bicycle rack
pedestrian - wheelchair	vehicle - car	

Table 3: List of classes in the NuScenes dataset.

from Table 3 including construction vehicles, construction workers, traffic cones and movable barriers. The number of annotations for the classes construction vehicle and construction worker was relatively high - 14671 and 9161 respectively [21].

3.2 Custom dataset

As discussed in the previous sections, the NuScenes dataset was selected as a starting point for the construction detection problem. For creating a custom dataset for this particular problem, all the candidate images which contained some form of construction work on the road were to be selected.

The nuScenes dataset was broken down into 10 parts for easy download and maintenance. Data from all the synchronized sensors was placed in each of the divided parts. Upon extracting and combining all the parts of the dataset, it had four folders -

- **samples** - sensor data for the keyframes.
- **sweeps** - sensor data for intermediate frames.
- **maps** - map related files.
- **v1.0-*** - JSON tables with all metadata and annotations.

For creating the custom dataset, the data had to be analyzed and filtered. As the number of images in the whole dataset was around 1.4 million camera images, processing the whole dataset was challenging. The initial idea was to create a proof of concept solution which is capable of detecting construction work with some baseline accuracy. Later, more data could be incorporated into the custom dataset to improve the reliability of the whole system. The images from the front camera were chosen as they would be more representative of images taken from a dashboard camera. A sample image from the dataset with construction work on the road is shown in Figure 18.

The first four parts were selected from the large dataset for manual classification. This initial manual classification and division of the dataset was done to make further

Figure 18: Construction work from the nuScenes dataset captured in Singapore with left hand traffic.

steps in the workflow easier. The amount of data was restricted to four parts of the entire dataset in order to meet the resource and time constraints.

This served as the base for creating the construction detection dataset. Although the NuScenes datset is focused on capturing the diversity in daily driving conditions across the globe, there are significant differences in colours and lighting conditions between images taken in Singapore (Figure 18), Boston (Figure 19) and Finland (Figure 20). The barricades and temporary structures placed on Finnish roads follow the yellow and red stripes pattern whereas the ones from Boston and Singapore have cones and barricades which are orange and white. To focus on this diversity and to make it possible for future expansion, images which were captured on Finnish roads were to be included in the dataset. A few manually captured images from around the city of Helsinki and images from a dashboard camera of a car were selected and included in the dataset. Although all practical scenarios cannot be represented, a general dataset with over 7,000 images from 3 cities of Boston, Singapore and Helsinki was created to address this problem. The processing of data and its labelling for classification and detection tasks are discussed in further sections.

3.3 Classification

A common way of maintaining an image classification dataset is to store the data in folders corresponding to the class names. This works as long as every image is associated with only a single class. This strategy is followed even in famous image classification datasets and competitions such as the imagenet challenge [5]. The images from the first four parts of the NuScenes dataset were manually classified into images containing construction work and images without construction work. This new dataset was organized similar to the imagenet challenge and placed into two folders corresponding to the two classes, one with construction work and another without the same. Like any classification dataset, the data was split into a training

Figure 19: Construction work from the NuScenes dataset captured in Boston with right hand traffic.

Figure 20: Image with construction work captured in Espoo, Finland.

and validation subsets and placed in two folders. The folder structure of the complete dataset is shown in Figure 21. The number of images in each class is shown in the same figure. The dataset contains 2072 images with some form of construction work on the road which was divided into training and validation subsets with a ratio of 75% : 25% at random. The images without construction work were also divided along the same lines. The dataset contains approximately twice the number of images without construction work as the number of images containing the same. 5831 images without construction work were included in the training directory and 1942 images for validation. The structure of the dataset and number of images in each class is also highlighted in Figure 21.

The following subsections discuss the selected deep learning algorithms used to address this binary classification problem.

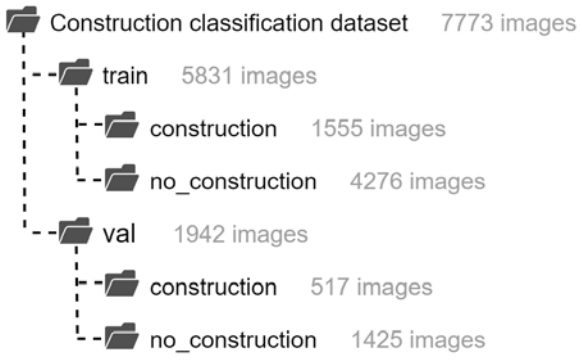

Figure 21: Directory tree for construction classification dataset.

Preprocessing

Pre-processing of the images is an important step in any deep learning application. For increasing the amount of variations in the training data, the data was augmented using a few predetermined transformations. The input images could be of varying resolution although the data taken from the NuScenes dataset has a standardized resolution of 1600×900, the images taken in Finland might have different resolutions based on the camera used to capture the image. Hence, a standard resolution was chosen around one fourth the standard resolution of a 1080p image. A pytorch resize transform was used for this purpose.

The pytorch transform for randomly flipping the image horizontally was used. This was done to train the model to be invariant towards changing driving directions in different countries. The quality of the data is also improved as there is more variation in the dataset. Blurring and other kinds of input noise were not introduced into the data during training.

3.3.1 Resnet

Resnet is one of the famous deep learning algorithms used for classification problems. It gained traction due to its impressive performance in the imagenet competition. Basically, it is a deep Convolution Neural Network (CNN) architecture used to solve complex computer vision problems such as object detection or image classification. To improve the performance of these models, the depth of the networks can be increased by incorporating more layers in the CNN. This creates more parameters to be learnt during the training phase. Some of the adverse effects of adding more layers to the network include the problem of vanishing or exploding gradients and degradation. While training the network, the weights of the network are updated based on a partial derivative of the error function with respect to the current weights at each iteration of the training. In some cases, this gradient will be vanishingly small and effectively prevent the weight from changing its value. On the other hand, in some cases, the gradients accumulate and result in large updates to the network weights and they increase uncontrollably. To address this problem of exploding and vanishing gradients, normalized initialization is used in the network. The network

also uses intermediate normalization layers which help the intermediate layers to converge during backpropagation. The degradation problem being referred to here involves degradation of the accuracy of the network as we increase the depth of the network. This happens once the accuracy gets saturated and optimization becomes increasingly difficult.

The Resnet architecture [22] tries to address these problems related to deep learning by introducing "identity mapping" or "shortcut connections" as shown in Figure 22. These identity mappings do not add any additional parameters or increase the training computational complexity and still can address the problem of vanishing gradients and degradation.

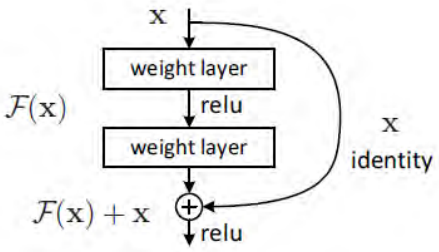

Figure 22: Resnet shortcut connections and basic building block [22]

Two networks based on the Resnet architecture were selected to be tested for the custom dataset created for this problem. To test the impact of the depth of the network based on the same architecture, Resnet 50 and Resnet 101 networks were selected. Resnet 50 is a deep learning network which is 50 layers deep. As discussed in Section 2.3.3, transfer learning was used and model already trained on the imagenet dataset was used as a starting point to improve the speed of training. PyTorch [23] was used for training and evaluating the network as it is one of the best frameworks for deep learning and widely used in the academic research work in recent years.

The imagenet dataset which involves classification of images into 1000 classes compares the performance of models based on certain error metrics which are relevant for multi-class classification problems. In the case of top-1 error rate, when the model makes prediction of probabilities of the input image belonging to a set of classes, if the one with the highest probability does not match with the true label, it is considered an error. On the other hand, in the case of top-5 error, the top 5 predictions of the model (sorted in the descending order of predicted probabilities) are compared against the true label of the input and if there is no match, it is considered an error. The top-1 and top-5 error metrics are computed as the ratio of the corresponding errors to the total number of input images.

A model based on the Resnet 50 architecture and trained on the imagenet dataset was used for the first experiment. This model had a top-5 error of 7.13% and a top-1 error of 23.85% [24]. This is one of the better performing models which we can adapt to the classification problem at hand. The pre-trained weights and the resnet-50 model were loaded into memory and slight modifications were made to the

network. The final fully connected layer which was used to classify the input into 1000 classes was removed and a fully connected layer with two outputs was created. They correspond to the two classes - one with construction work and another without construction work.

For training the models on the custom construction classification dataset, cross-entropy loss was minimized to obtain a more optimal model. As discussed in Section 2.2.1, this serves as a good objective function during the training of the classifier.

The training data was divided into batches based on the memory capacity of the GPU used on the training system. Metrics of accuracy and loss were logged for each epoch to identify the models best stable performance beyond which there was negligible improvement. Other metrics which were evaluated at each epoch include the metrics of the confusion matrix as discussed in Section 2.2.1.

Ten experiments were conducted with the Resnet-50 model with different values for the parameters such as learning rate, momentum and epochs. The experiment with the best performance was selected.

The training and validation loss and accuracy plots are shown in Figure 23 and Figure 24.

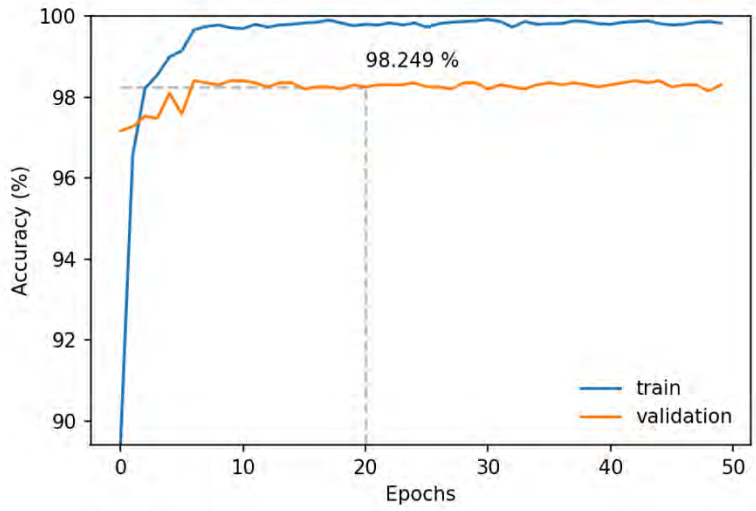

Figure 23: Plot of Resnet-50 model loss during training and validation.

		Actual Class	
		Construction	No construction
Predicted Class	Construction	**499**	16
	No construction	18	**1409**

Table 4: Confusion matrix for Resnet-50 based network.

The Resnet-50 based model was able to achieve an accuracy of 98.25% after training for around 20 epochs. The training was stopped around 50 epochs. The

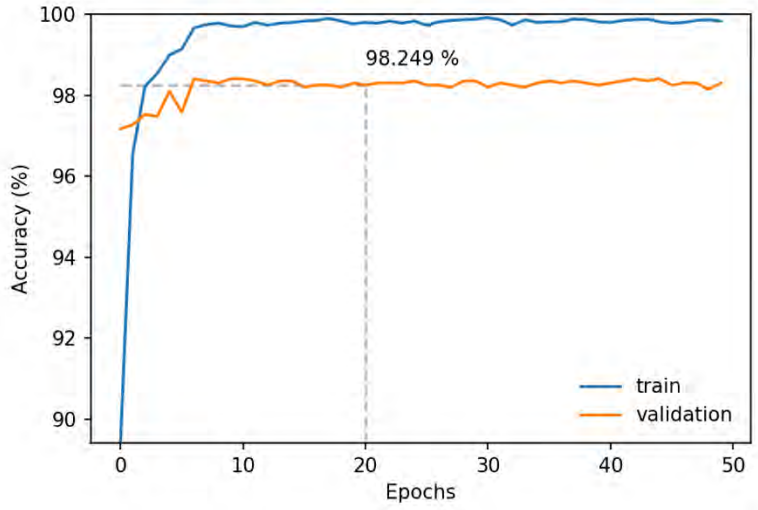

Figure 24: Plot of Resnet-50 model accuracy during training and validation.

performance of this model is summarized in Table 4. As seen in the confusion matrix, the model was able to successfully classify 499 images out of 515 images which contained some form of construction work on the road. The false positives are also relatively low at 18.

A second model based on the Resnet architecture was selected for analyzing the effect of a deeper network on the same dataset. The Resnet-101 network was chosen for this experiment. The pre-trained model used for this network was also trained on the imagenet dataset and had a top-5 error rate of 6.44% and a top-1 error rate of 22.63% [24]. Similar modifications were made to the final fully connected layer of this network and the 1000-output layer was replaced by a fully connected layer with two outputs corresponding to the two classes. Again, cross-entropy was used as the loss function to be minimized. The results of the best performing model on this dataset is summarized in Table 5. The plots of accuracy and loss for the training of this model is shown in Figure 25 and Figure 26.

		Actual Class	
		Construction	No construction
Predicted Class	Construction	**502**	16
	No construction	15	**1409**

Table 5: Confusion matrix for Resnet-101 based network.

3.3.2 Densenet

Densenet was a network developed by a collaborative effort between researchers from Cornell University, Tsinghua University and Facebook AI Research (FAIR) groups in 2018 [25]. Densenet is another CNN which can be leveraged for solving classification

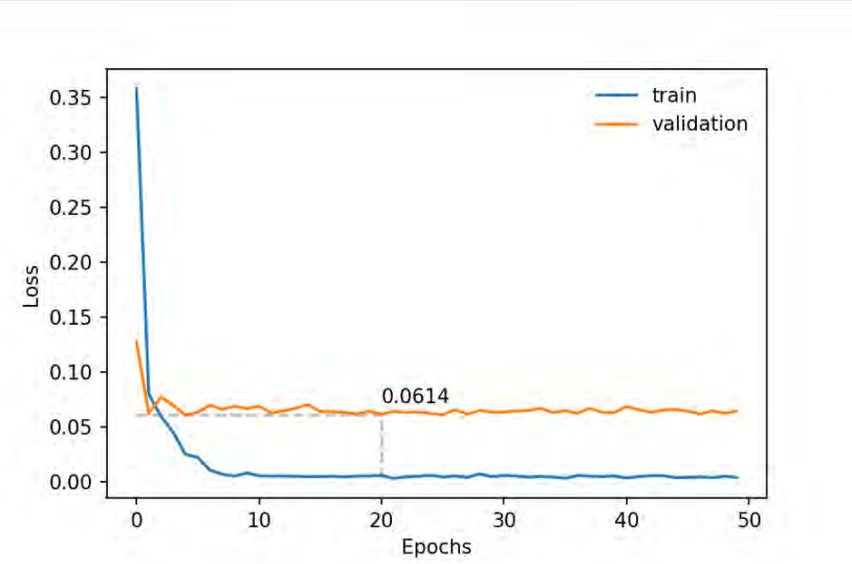

Figure 25: Plot of Resnet-101 model loss during training and validation.

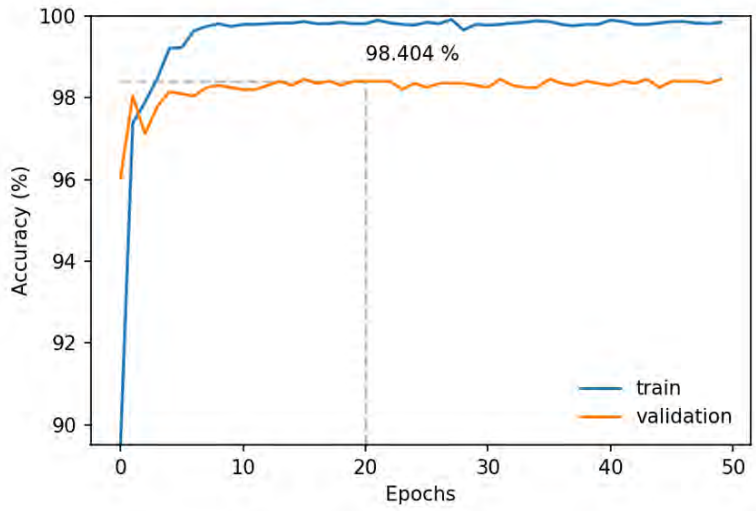

Figure 26: Plot of Resnet-101 model accuracy during training and validation.

and detection problems. The central feature of densenet is the dense block which consists of a sequence of fully connected layers with matching feature-map sizes. The authors propose that this ensures maximum information flow between the layers. The layout of connections can be schematically shown as in Figure 27. The key feature of this architecture is that each group of layers passes its output to all further layers and receives inputs from all the preceding layers.

In theory, each layer is receiving a "collective knowledge" from all the preceding layers [25]. As each layer of the network is receiving inputs from all the preceding layers, the network can be created to be more compact or thinner by having lesser number of channels. Hence, a higher computational and memory efficiency can be

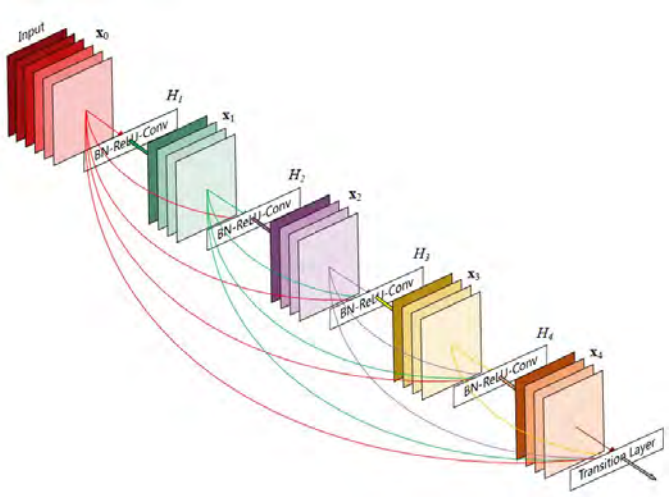

Figure 27: 5-layer dense block. [25]

achieved. This design yields another interesting benefit and acts as a way to counter the problem of vanishing gradients. The error can be easily propagated from the final classification layer to the earlier layers in a more direct fashion.

A pre-trained model based on the Densenet-161 architecture was selected for this experiment. The model was trained on the imagenet dataset and had a top-5 error of 6.2% and a top-5 error rate of 22.35% [24]. This model was selected for its impressive performance. The deeper network has a better top-5 error rate than the models discussed in the previous sections. While training the densenet model on the custom classification dataset, cross-entropy loss was used as the objective function. The results of the best performing model on this dataset is summarized in Table 6. The plots of accuracy and loss for the training of this model is shown in Figure 28 and Figure 29.

		Actual Class	
		Construction	No construction
Predicted Class	Construction	**503**	16
	No construction	14	**1409**

Table 6: Confusion matrix for Densenet-161 based network.

3.4 Object detection

Storing object detection datasets is not as straightforward as classification datasets as more information regarding each bounding box needs to be saved along with the type of object present in the bounding box. Each image might contain a single or multiple bounding boxes corresponding to different object classes. Various object detection related competitions use their standards for storing these object annotations. Some of the famous formats include the PASCAL VOC, YOLO and MS COCO formats. All

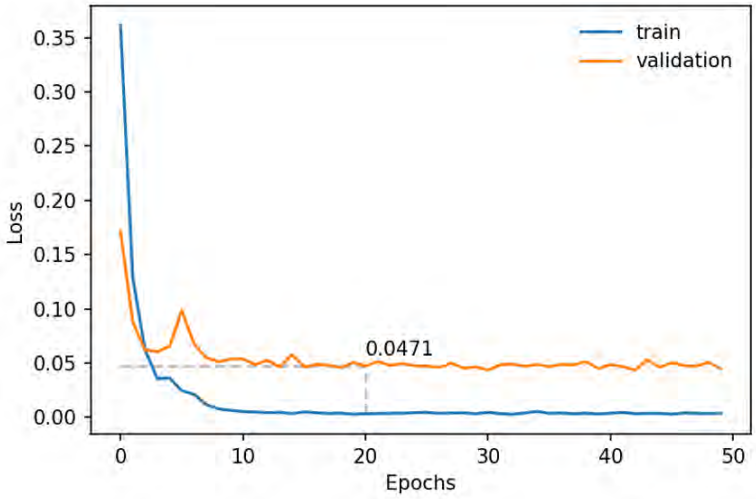

Figure 28: Plot of Densenet-161 model loss during training and validation.

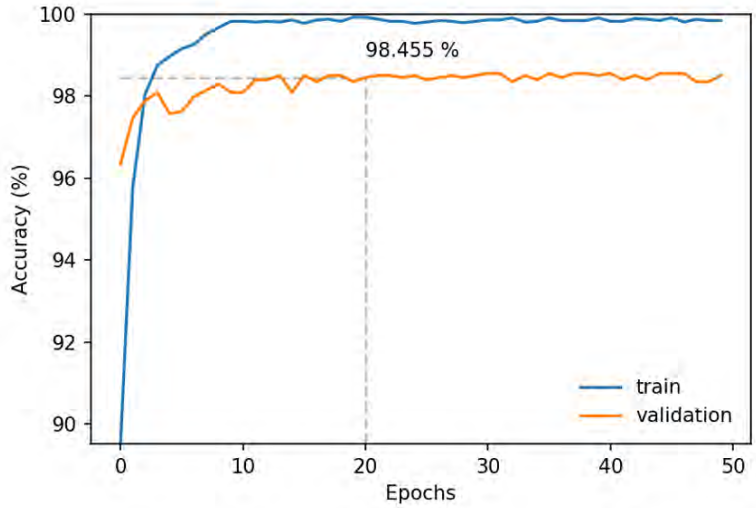

Figure 29: Plot of Densenet-161 model accuracy during training and validation.

these annotation formats store the same basic information related to the bounding box position and class along with some additional data.

The YOLO format stores the annotations in separate files corresponding to each image. Each annotation file contains one line for each bounding box. Each line has five parameters, the first one is a number corresponding to the object class. The next set of numbers correspond to the relative x-centre, y-centre, width and height of each bounding box. These numbers are evaluated relative to the image size, i.e. the coordinates of the centre, the width and height of the bounding box are measured relative to the total image width and height. Hence, all numbers range from 0 to 1.

The MS COCO format tries to address this problem in a slightly different way.

The annotations for all the images are saved in a single file. The training, validation and testing sets would have different annotation files. The COCO format also provides a means for saving annotations for other problems such as segmentation and captioning in the same file. The annotations are saved in JSON format keeping future expansion in mind.

Some of these annotation formats use JSON while others use XML or plain text and vary in the amount of information being saved. The machine learning algorithm needs to be able to parse the inputs and process them accordingly. A generic labelling tool was used for creating annotations for the image dataset and is discussed in the following section.

3.4.1 Labelling tool

The OpenCV group created a labelling tool called CVAT (Computer Vision Annotation Tool) for annotating images and video for object detection, and segmentation [26]. This tool provides an easy graphical web-based interface for annotation. The interface can be run locally on any machine with a browser installed. Local datasets can be loaded for processing on the browser.

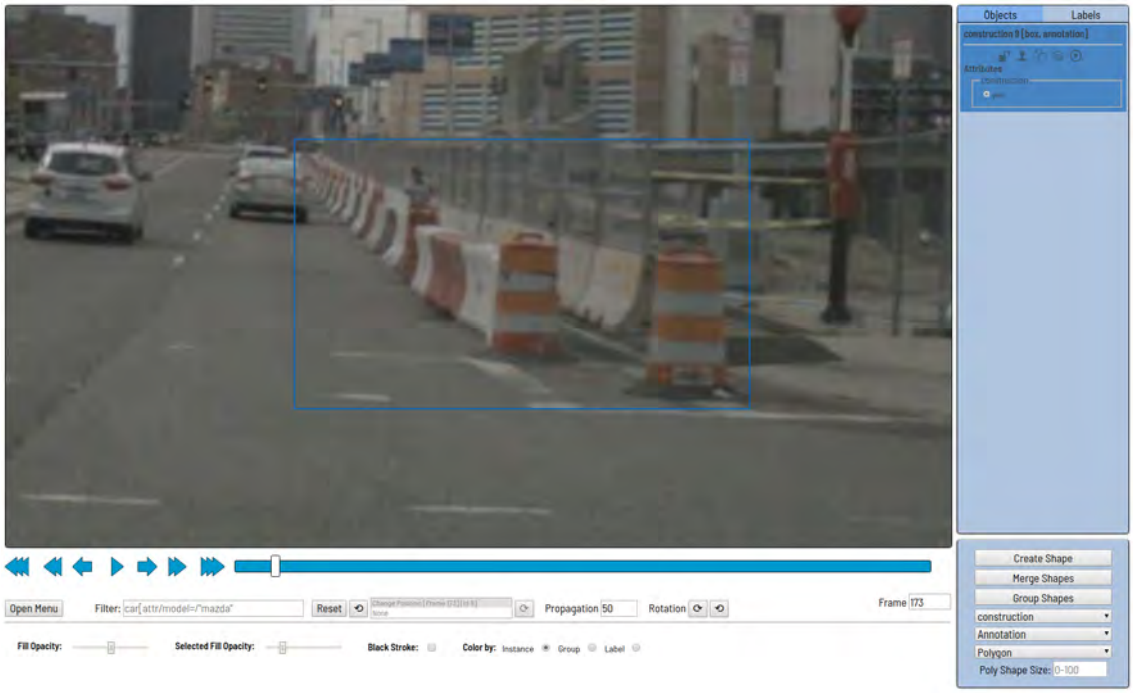

Figure 30: A view of the annotation interface from CVAT.

The tool can be used to create different types of polygons and assign corresponding class labels to each one. The classes and parameters need to be defined and the images for the dataset need to be selected before launching the labelling interface. A view of the CVAT interface is shown in Figure 30. The created annotations can be exported in various standard formats such as the YOLO format, MS COCO format, Pascal VOC format and many more as per the necessity of the problem and

algorithm being used. At any point, the annotations for the dataset can be imported and exported from the user interface for creating checkpoints during the annotation process.

3.4.2 Method 1 - dataset for detection

As the current task is to localize the construction work within the image, bounding boxes are used instead of more complex polygons. As we get into more complex shapes or change to problems such as segmentation, the complexity of the problem and the model increases significantly.

The next task was to identify and define meaningful labels to accurately distinguish between the classes which were going to be annotated. This task was addressed by two different methods, the first option was to detect and classify construction sites as a whole, and the second method involved detecting individual components of what constitutes a construction site. These elements could include objects such as construction vehicles, construction workers, traffic cones, and temporary barricades.

For the first method, after analyzing numerous variations of images containing road construction work, there was a drastic visual variation of what constitutes a road construction work. Hence, two broad classes were selected for annotation as follows -

- **constr_type_1** - These type of annotations include temporary barricades and arrangements of traffic cones to partially or completely block road traffic.
- **constr_type_2** - This type of construction annotations include active construction sites with construction workers and vehicles working on the road.

A few sample images containing both these types of construction work is shown in Figure 31 and Figure 32. As seen from these samples, the presence of excavators or heavy machinery along with a lot of workers on the scene creates a drastic difference in visuals of what constitutes the construction site.

To create the dataset for the detection task, images were taken from the previous classification dataset (Section 3.3) which contained images with construction work and the CVAT tool was used to create annotations for the two classes.

The newly created dataset for the detection task contained **554 unique annotations** belonging to the **'constr_type_1'** class and **533 unique annotations** of the **'constr_type_2'** class. Negative images without any bounding boxes were added from the images without construction work from the classification dataset. To create the training and validation parts of the dataset, the 1087 images containing construction work were divided in the ratio of 4:1 and negative images without any construction work were added to each part.

The YOLO v3 network was used as the object detection model. The model is discussed briefly in the following section.

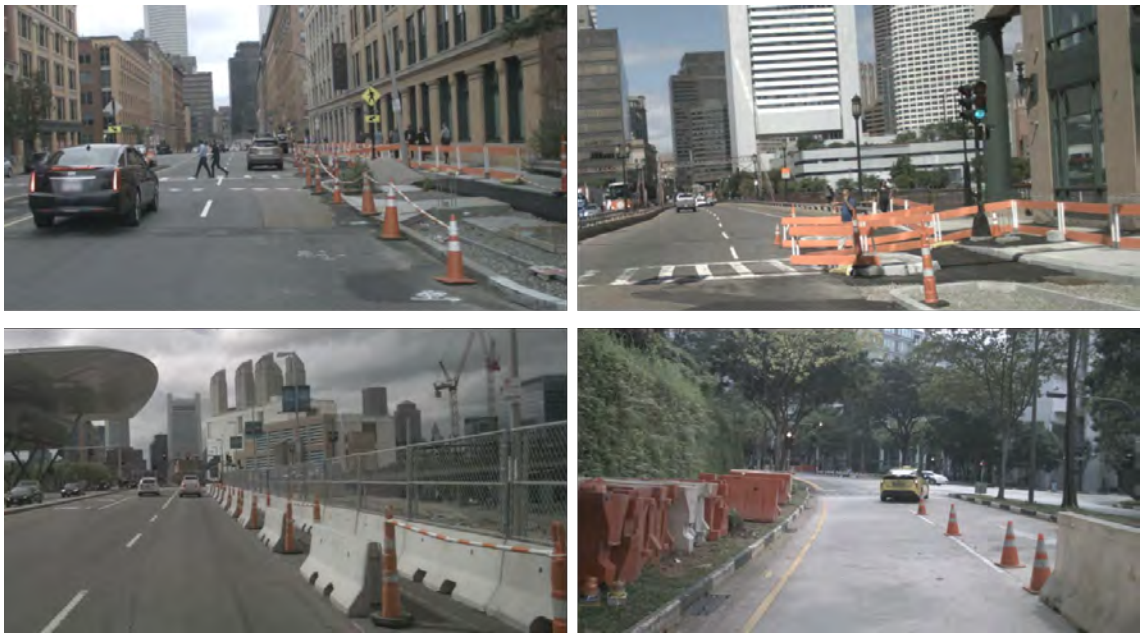

Figure 31: Sample images with constr_type_1.

Figure 32: Sample images with constr_type_2.

3.4.3 Yolo V3

YOLO V3 [27] is a classifier and localizer network used for object detection tasks. The algorithm uses a single convolution based network for looking at the whole image and predicting bounding boxes and probabilities. This makes the network faster than other famous networks such as R-CNN and Fast R-CNN [28] which use region proposals of different sizes and a classifier to detect objects in images. The network

consists of a total of 106 convolutional layers and is invariant to changes in input image size.

Behind the scenes, the YOLO algorithm uses logistic regression for bounding box prediction while minimizing sum of squared error loss. The algorithm uses binary cross-entropy loss with independent logistic classifiers for the class predictions. The authors of the YOLOV3 algorithm have also refrained from selecting the class assigned to the detected bounding box using the softmax function. This function assumes that all classes are mutually exclusive from each other by assigning the class with the highest probability to the detected bounding box. Instead, the authors identify the presence of a class if the confidence score is higher than the predetermined threshold for the classes. In the current problem, the classes are not completely mutually exclusive and there is a slight overlap between the classes. Although there are drastic changed between what constitutes a construction site of type 'const_type_1' and 'const_type_2', the barricades remain a similar feature present in both the classes. Due to the working principles and the proven performance of the YOLOv3 algorithm, it would be a great fit for the current problem.

3.4.4 Results

As discussed before, the YOLO algorithm uses annotations in a specific format, each image file should be accompanied by an annotations file with the same name as the image and a '.txt' extension. The annotations created for the dataset using the CVAT tool were converted to this format and divided into training and validation datasets. Two text files were created to list the training and validation file names. The network was configured to predict two object classes as described in Section 3.4.2. Pretrained weights from a network trained on the imagenet dataset was used to initialize the weights for the current problem. [28].

Experiments were conducted using different parameters such as learning rates, momentum, augmentation, epochs and resolution. The results of the experiment with the best mAP score is highlighted in Figure 33.

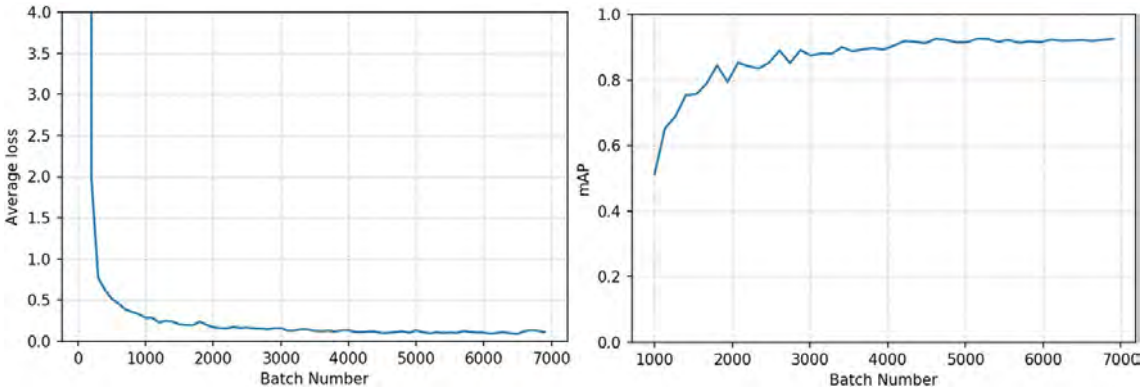

Figure 33: Average loss and mean average precision while training YOLOv3 on the custom detection dataset.

The network was trained with a learning rate of 0.001. The momentum was set

at 0.9 which is indicative of how much the history of the training affects the further change of weights. The burn-in was set at 2000 epochs during which the learning rate gradually increases till the target learning rate of 0.001. The best mAP score achieved on this dataset by the YOLOv3 network was 0.92568 at a 50% IoU. An example of a correct detection of construction work of type 2 is shown in Figure 34. Another case when the model closely identified the construction work but failed to correctly classify the type is shown in Figure 35.The following section highlights the experiments with the second approach for the detection problem.

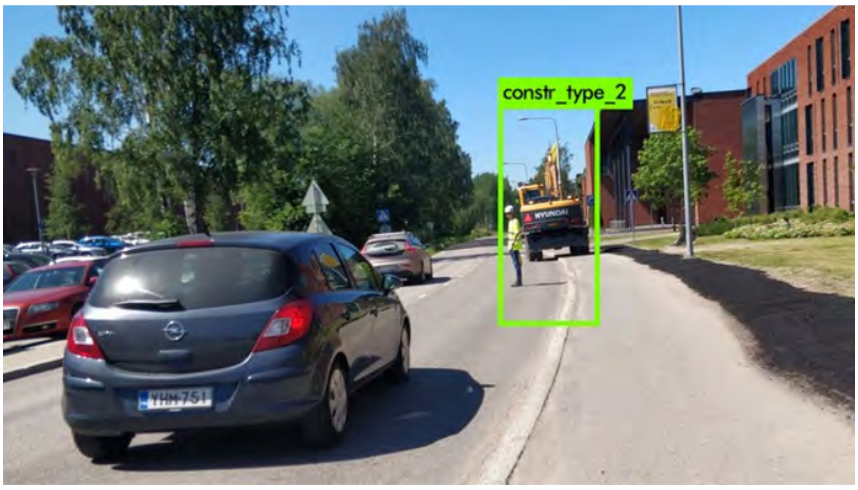

Figure 34: True Positive construction work detected by the trained YOLO V3 algorithm.

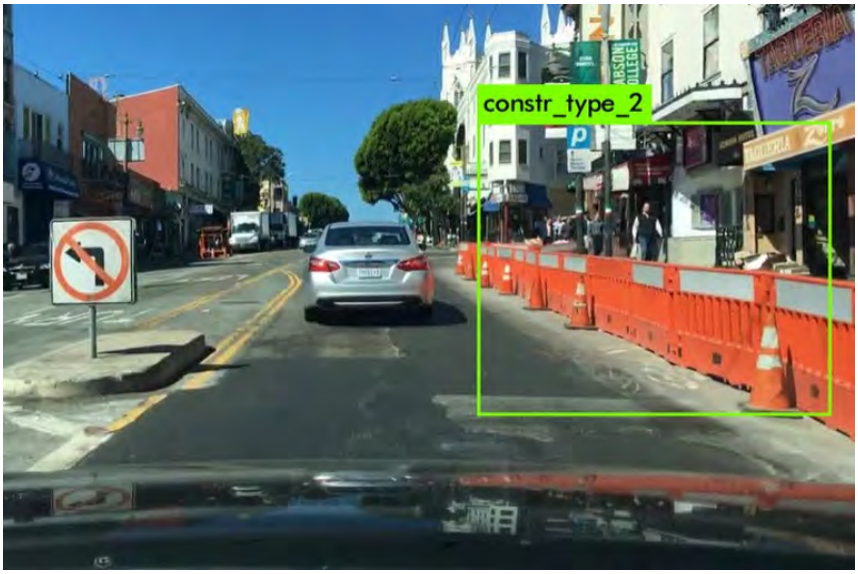

Figure 35: Construction work detected and wrongly classified as type 2 by the model.

3.4.5 Method - 2 Using the NuScenes dataset

The goal of the second approach is to identify the commonly occurring elements of what constitutes a construction site. Detecting these smaller components would give us further insight and confidence as to whether a construction site is present in a particular image. The NuScenes dataset has some interesting relevant classes which have already been annotated. These classes include barriers, construction vehicles, construction workers, and traffic cones. The dataset provides 3D bounding box annotations for the classes. These annotations were translated into 2D bounding boxes on to corresponding images and exported to a JSON file. The annotations for the four selected classes were filtered out from the other classes for creating the dataset for this experiment. As the NuScenes dataset is a very large image dataset, the training and evaluation of models for this dataset took considerably more time than the experiments with the previous dataset.

Networks based on two different architectures were used for establishing a preliminary baseline performance on this dataset. The first network is based on the YOLO architecture and the second network is based on the Retinanet architecture. A summary of the latter is provided in the following section.

3.4.6 Retinanet

Retinanet [29] is another one-shot detector algorithm based on the concept of 'focal-loss' developed by the Facebook AI Research group. The focal loss concept prevents the network from getting overwhelmed from the class imbalance and a large number of negatives in the dataset. The authors of the algorithm have proven that the network performs better on the MS COCO dataset than the Faster R-CNN network while being a single stage detector. Hence this network was chosen to have a comparative result to the YOLO model.

3.4.7 Results

The YOLO V3 based network was configured to predict four classes and the annotations were converted to the format supported by the algorithm. The model was initialized with weights which were pretrained on the imagenet challenge.

Similarly, for the model based on retinanet, the network was configured to have four outputs corresponding to the four classes, and the annotations were converted to the format accepted by the algorithm. The model was initialized with weights which were pretrained on the MS COCO dataset. The dataset was divided into training and validation sets and training of both networks on the dataset were carried out.

The results are summarized in Table 7. The mean Average Precision obtained for each class is listed in the table against the algorithm used. The overall mAP at 50% IoU for the YOLO V3 based network was 30.53% and the retinanet based model was able to achieve a mAP of 33.39%. Although the two models have close overall mAP scores, the retinanet model performs much better for detecting construction vehicles and workers than the YOLO model.

Class	YOLO V3	Retinanet
Barrier	40.57	32.04
Construction vehicle	18.46	42.05
Construction worker	24.93	32.69
Traffic cones	38.16	34.07

Table 7: Class-wise mAP values for the two networks at 50% IoU.

A certain amount of error is introduced into this system during the conversion of the 3D annotations to 2D annotations. This could have played a small role as to why the accuracy of the models is relatively low. The 3D annotations were created using additional sensor data from lidar and radar sensors. When the images were captured by the camera on the car, the radar might have seen an object which might not have been in the direct line of sight from the camera's point of view. The script to convert the annotations does not consider this occlusion while translating the 3D annotaiotns onto the images. This is one of the drawbacks of this method of converting annotations. The goal of the current work was to use cameras for detecting construction work. Hence, the system should not rely on sensors such as lidar and radar. Recreating annotations for this huge dataset in 2D would require a huge investment of effort and cost which is beyond the scope of the current work.

The following section discusses concluding thoughts and how this work could be incorporated into future solutions.

4 Conclusion and Future Work

The goal of this thesis work was to explore approaches to vision-based detection of construction work on roads using images taken from a single camera such as a dashboard camera. This would be helpful for autonomous driving and navigation solutions once the construction work has been identified and localized.

This work focused on creating a foundation for building solutions for this vast problem by creating datasets and exploring deep learning based computer vision solutions. The results of training on the dataset is not fully tuned and optimized, but rather, they provide a baseline performance on the corresponding datasets and a proof of concept for future expansion and fine tuning. The problem was addressed form a bottom-up approach starting from an image classification problem and increasing the complexity to an object detection and localization problem.

For the binary classification problem, images from different geographical regions were assimilated for creating a diverse dataset which covered variations of what constitute a construction site. The NuScenes dataset was used as a starting point for creating this dataset along with additional images captured in Finland. Deep learning based algorithms using Resnet and Densenet were trained on the dataset and their performance was quantitatively evaluated. The model based on the Resnet-50 architecture provided a good balance between performance and lower computational complexity compared to the Densenet-161 and Resnet-101 models.

For the object detection and localization problem, the previous dataset was extended with custom annotations of two types of construction work and a model based on the YOLO V3 architecture was trained on this dataset. Although manual annotation is a time consuming process, the algorithm was able to identify construction work with reasonable reliability. An additional method was explored using annotations from the NuScenes dataset related to construction work. Algorithms based on YOLO V3 and Retinanet architectures were tested and their performance was evaluated as discussed in Sections 3.4.7.

The results form the first approach of binary classification of construction along with the two detection algorithms could be used to create a voting based algorithm to identify construction sites with more confidence.

This research work has potential to be expanded into future work related to autonomous driving. Construction work detection could be integrated into the existing dashboard camera systems in modern automobiles and this crowd sourced data could be used to update a real time map with live construction work happening in any region. Another possible future expansion to this work could be identifying the extent to which the construction work is blocking the driving lanes. Driving assistance systems or autonomous systems could use this additional information to plan or navigate the area around the construction work safely.

This work provides a good starting point for construction work detection and localization using only images. The design was created keeping crowd-sourcing and future expansion in mind.

5 References

[1] S. T. Barnard and M. A. Fischler, "Computational stereo," ACM Computing Surveys (CSUR), vol. 14, no. 4, pp. 553–572, 1982.

[2] I. Goodfellow, Y. Bengio, and A. Courville, Deep Learning. MIT Press, 2016, http://www.deeplearningbook.org.

[3] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," Proceedings of the IEEE, vol. 86, no. 11, pp. 2278–2324, 1998.

[4] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in 2009 IEEE conference on computer vision and pattern recognition. IEEE, 2009, pp. 248–255.

[5] "Imagenet summary and statistics," http://www.image-net.org/about-stats, [Accessed: 10 October 2019].

[6] M. Everingham, L. Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," Int. J. Comput. Vision, vol. 88, no. 2, p. 303–338, Jun. 2010. [Online]. Available: https://doi.org/10.1007/s11263-009-0275-4

[7] T. Lin, M. Maire, S. J. Belongie, L. D. Bourdev, R. B. Girshick, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: common objects in context," CoRR, vol. abs/1405.0312, 2014. [Online]. Available: http://arxiv.org/abs/1405.0312

[8] S. J. Pan and Q. Yang, "A survey on transfer learning," IEEE Transactions on knowledge and data engineering, vol. 22, no. 10, pp. 1345–1359, 2009.

[9] S. Dipanjan, "A comprehensive hands-on guide to transfer learning with real-world applications in deep learning," https://towardsdatascience.com/a-comprehensive-hands-on-guide-to-transfer-learning-with-real-world-applications-in-deep-learning-212bf3b2f27a, [Accessed: 18 August 2019].

[10] A. Wimmer, T. Jungel, M. Glück, and K. Dietmayer, "Automatic generation of a highly accurate map for driver assistance systems in road construction sites," in 2010 IEEE Intelligent Vehicles Symposium. IEEE, 2010, pp. 281–286.

[11] P. Kunz and M. Schreier, "Automated detection of construction sites on motorways," in 2017 IEEE Intelligent Vehicles Symposium (IV). IEEE, 2017, pp. 1378–1385.

[12] W. Ji, L. Tang, D. Li, W. Yang, and Q. Liao, "Video-based construction vehicles detection and its application in intelligent monitoring system," CAAI Transactions on Intelligence Technology, vol. 1, no. 2, pp. 162–172, 2016.

[13] F. Bröker, "Construction site detection for autonomous vehicles using deep learning," Master's thesis, Freie Universität Berlin, 2017.

[14] B. Zinkel, "Frustrating the future: How autonomous vehicles are vexed by construction," 2018. [Online]. Available: https://www.tsasafety.com/blog/frustrating-future-autonomous-vehicles-vexed-construction/

[15] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," The International Journal of Robotics Research(IJRR), vol. 32, no. 11, pp. 1231–1237, 2013.

[16] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 3213–3223.

[17] G. Neuhold, T. Ollmann, S. Rota Bulo, and P. Kontschieder, "The mapillary vistas dataset for semantic understanding of street scenes," in Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 4990–4999.

[18] "Mapillary platform," https://www.mapillary.com/, [Accessed: 10 August 2019].

[19] F. Yu, W. Xian, Y. Chen, F. Liu, M. Liao, V. Madhavan, and T. Darrell, "Bdd100k: A diverse driving video database with scalable annotation tooling," arXiv preprint arXiv:1805.04687, 2018.

[20] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, "nuscenes: A multimodal dataset for autonomous driving," arXiv preprint arXiv:1903.11027, 2019.

[21] "Nuscenes data annotation," https://www.nuscenes.org/data-annotation, [Accessed: 15 August 2019].

[22] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 770–778.

[23] "Pytorch." [Online]. Available: https://pytorch.org/

[24] "torchvision.models." [Online]. Available: https://pytorch.org/docs/stable/torchvision/models.html

[25] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 4700–4708.

[26] "Computer Vision Annotation Tool (CVAT)," https://github.com/opencv/cvat/, [Accessed: 10 October 2019].

[27] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," arXiv preprint arXiv:1804.02767, 2018.

[28] "Yolo: Real-time object detection," https://pjreddie.com/darknet/yolo/, [Accessed: 10 October 2019].

[29] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in Proceedings of the IEEE international conference on computer vision, 2017, pp. 2980–2988.