Northeastern University

Improving NRCan - ViaRail maintenance using Computer Vision

Final Report

Kishan Patel

Dr. Michal Aibin

Dr. Yvonne Coady

December 18, 2021

1 Introduction

Object Detection is an elemental part of Computer vision with a wide range of real-world applications. It involves the detection of various objects in digital images or video. One such object detection algorithm is Yolo (You only look once) which has gained much popularity in the computer vision community due to its high performance and low inference time. In this project, we propose a proof of concept of how the Yolo algorithm can be leveraged to improve the maintenance of railway tracks operated by Via Rail Canada. Via rail operates approx 500 trains running on 12,500 km of track. These tracks pass through long stretches of sparsely populated lands which makes maintaining these tracks a tough job due to the sheer amount of resources required to identify the points of interest such as Vegetation, Missing or broken Tie, etc. This project uses the Yolo algorithm to identify these points of interest with the help of drone flights. This project also explores the performance of different versions of Yolo (Yolov4 Yolov5) to achieve the highest accuracy possible.

2 Related Works

It is impractical to assume that images are always captured in perfect light conditions. Low light images usually degrades with scene height and suffers from severe object information loss. Author in article(6) proposed physical based image enhancement enhances the image constrast by exploiting the relation among the atmosphere light and transmission map. Another subnet detects salient object from enhanced images. Author Used five SOD datasets such as DUT-OMRON, PASCAL-S, NTI-V1 are used to evaluate model for precision recall curves, F-measures and Mean absolute error. Promising results were observed in NTI-V1 and public dataset with high volume of false postive due to noise interference.

Large number of images with ground truth information in form of annotation is bottleneck for any object detection application. Author in article(1) studies existing crowdsource techniques such as Amazon Mechanical Turk and proposed turn based annotation system consisting of three simple task (single object detection, a quality verification task, coverage verification task) is proposed. Pascal VOC 2007 dataset was used for evaluation where 3 rounds of labeling are performed to deal with low annotation difficulty. Proposed technique shows improvement in accuracy by up to 8-10%. Proposed system can assist object detector to obtain higher accuracy in a cost effective manner.

Design a lightweight model that can balance the computational complexity and accuracy simultaneously. A neural network model named Embedded YOLO in article (5) which reduces number of parameters by using DSC_CSP module to replace the middle layers of YOLOv5. Author used IVSLab and BDD100k dataset to evaluate performance of embedded Yolo architecture based mAP, model size, GLOPs and speed. Model achieved 0.59 mAP with 41 FPS on the mediaTek's Dimensity 1000 platform which confirmed suitability of proposed model on embedded systems.

To improve the real-time of object detection, a fast object detection method on YOLOv4 based on YOLOv4-tiny to simplify the network structure and reduce parameters was proposed in article (2), which makes it suitable for developing on the mobile and embedded devices. they replaced two ResBlock-D modules with two CSPBlock modules in Yolov4 Tiny to reduce computation complexity also implemented auxiliary residual network block to extract more

feature information of object to reduce detection error. The mAP (mean value of average precision) , FPS (Frames per second) and GPU utilization was used to quantitatively evaluate the performance of different methods compared with YOLOv3, YOLOv4, YOLOv3-tiny, YOLOv4-tiny to test their performance in mAP and FPS. Simulation results show that the proposed method has faster object detection than YOLOv4-tiny and YOLOv3-tiny, and almost the same mean value of average precision as the YOLOv4-tiny.

State-of-the-art object detection networks depend on region proposal algorithms to hypothesize object locations. Advances like SPPnet and Fast R-CNN have reduced the running time of these detection networks, exposing region proposal computation as a bottleneck. Authors in article (3) introduced a Region Proposal Network (RPN) that shares full-image convolutional features with the detection network, thus enabling nearly cost-free region proposals. An RPN is a fully-convolutional network that simultaneously predicts object bounds and objectness scores at each position. RPNs are trained end-to-end to generate highquality region proposals, which are used by Fast R-CNN for detection. They comprehensively evaluate their method on the PASCAL VOC 2007 detection benchmark. For the ImageNet pre-trained network, they use the "fast" version of ZF net that has 5 conv layers and 3 fc layers, and the public VGG-16 model5 that has 13 conv layers and 3 fc layers. they primarily evaluate detection using mean Average Precision (mAP), because this is the actual metric for object detection. detection system had a frame rate of 5fps (including all steps) on a GPU, while achieving state-of-the-art object detection accuracy on PASCAL VOC 2007 (73.2% mAP) and 2012 (70.4% mAP) using 300 proposals per image.

Object detection is a fundamental task in computer vision with wide application prospect. However, most algorithms suffer from high computation cost and long inference time, which makes them impossible to be deployed on embedded devices in real industrial application scenarios. Authors in article (7) propose the Mobile CenterNet to solve this problem. Their method is based on CenterNet but with some key improvements. To enhance detection performance, They adopt HRNet as a powerful backbone and introduce a category balanced focal loss to deal with category imbalance problem. knowledge distillation is employed to transfer knowledge from cumbersome model to a compact one. They conduct experiments on a large traffic detection dataset BDD100K and validate the effectiveness of all the modifications. Their method achieves the 1st place in the Embedded Deep Learning Object Detection Model Compression Competition held in ICME 2020 with 44.6 mAP.

Model efficiency has become increasingly important in computer vision. In this paper (4), they systematically study neural network architecture design choices for object detection and propose several key optimizations to improve efficiency. First, They propose a weighted bi-directional feature pyramid network (BiFPN), which allows easy and fast multi-scale feature fusion; Second, They propose a compound scaling method that uniformly scales the resolution, depth, and width for all backbone, feature network, and box/class prediction networks at the same time. Developed a new family of object detectors, called EfficientDet, which consistently achieve much better efficiency than prior art across a wide spectrum of resource constraints. with single-model and single-scale, their EfficientDetD7 achieved state-of-the-art 52.2 AP on COCO test-dev with 52M parameters and 325B FLOPs1 , being 4x – 9x smaller and using 13x – 42x fewer FLOPs than previous detector.

Accurate real-time object detection plays a key role in various practical scenarios such as automatic driving and UAV surveillance. The memory limitation and poor computing power of edge devices hinder the deployment of high performance Convolutional Neural Networks (CNNs). Iterative channel pruning is an effective method to obtain

lightweight networks. In this paper (8), to measure the channel importance, they simultaneously consider the scale factor of batch normalization (BN) and the kernel weight of convolutional layers. Generous ablation studies are imported to investigate the properties of proposed method and compare it with similar state-of-the-art algorithms. Pruned-YOLOv3/v5, which is constructed via pruning YOLOv3/v5. The experimental results on the MS-COCO and VisDrone datasets show that the proposed model achieves a satisfactory balance between computational efficiency and detection accuracy.

3 Research Questions

Based on information gained in the literature studies and project objectives, following research questions were proposed for this research study.

- **Q1:** How can we leverage Image manipulation to enhance the annotation process?
- **Q2:** How can we leverage different augmentation techniques to signify the points of interest?
- **Q3:** How can we leverage different versions of the Yolo object detection algorithm to achieve higher accuracy in detecting the points of interest?

4 Methodology

Following Figure 1 shows the model pipeline with each of the steps involved from data gathering to running inferences on the images.

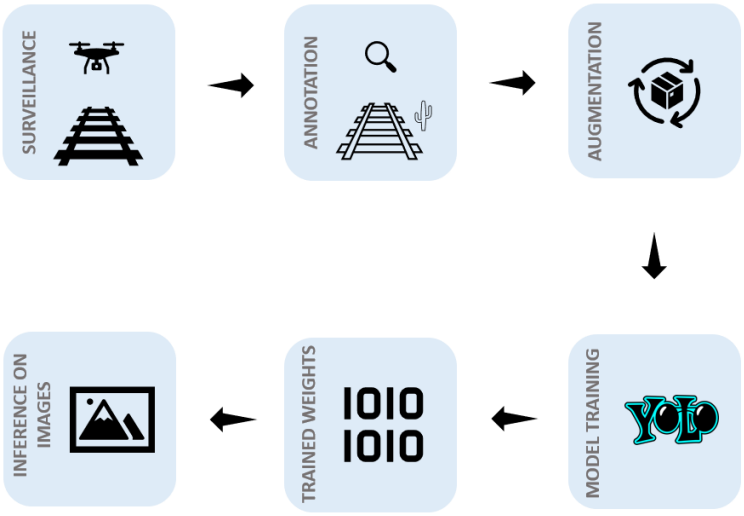

Figure 1: Model Pipeline

4.1 Dataset Pre-processing

Dataset pre-processing is performed to ensure that the data is in the suitable format for the algorithm's training. The initial dataset was comprised of a total of 395 images showing tracks from various stretches of railway. After an initial assessment, the following pre-processing steps were performed on the dataset to enable efficient annotation, which is critical for machine learning model performance.

4.1.1 Image Orientation Correction

Annotation is performed with a square or rectangular block signifying the point of interest. To make the annotation process more manageable and efficient, the orientation of images was corrected such that tracks are in a vertical or horizontal direction.

4.1.2 Artificial Vegetation

As the rail tracks are well maintained, there were very few images where vegetation was visible. To achieve higher accuracy, we needed to train the model using a well-balanced set of images with points of interest; thus, artificial vegetation was added on and besides the tracks.

4.1.3 Artificial Missing Ties

Similarly, there were no images with missing ties; thus, ties were removed from some of the images to train the model for the missing tie scenarios.

4.2 Annotation

After pre-processing the data, the next step was to start annotating the images for training the algorithm. 4 points of interest that were selected for the training purpose were as follows:

- **Vegetation**: Each image was reviewed closely to check any vegetation near the track. If it was present at a close distance to the track, it was annotated so that algorithm gets trained accordingly.
- **Missing Tie**: The portion of the track was annotated where there was a missing tie.
- **Broken Tie**: If there was a crack on the tie or seemed broken, it was annotated as a broken tie.
- **Others**: Any unidentified object on the tracks was classified as others.

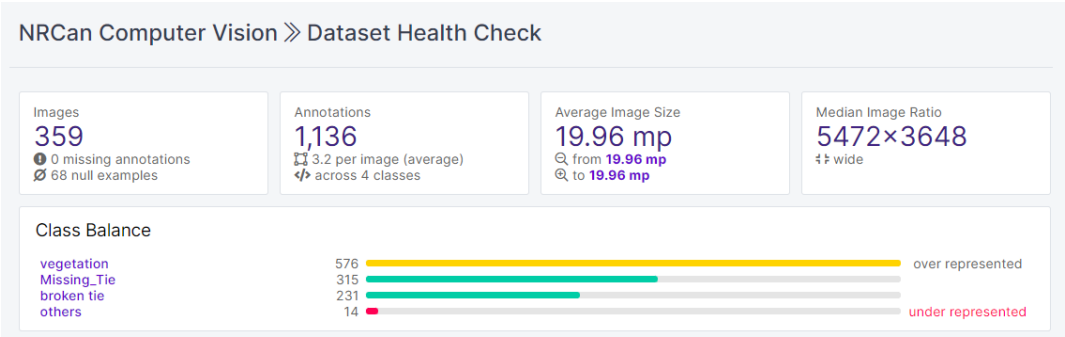

Figure 2: Dataset Health Check

Figure 2 shows the overview of the dataset. There were 359 images in the dataset, including the images containing missing or null annotations. There were 1,136 objects annotated in 359 images, out of which 576 was vegetation, 315 was missing tie, 231 were broken tie, and 14 were the others.

5 Evaluation

5.1 Simulation Setup

Two versions of the dataset were created for different versions of Yolo to observe the performance differences. Saturation augmentation was applied to both the versions which describes the depth or intensity of colour present within an image. Saturation is also referred to as Chroma. The more saturated an image is, the more colourful and vibrant it will appear. Less colour saturation will make an image appear subdued or muted.

5.1.1 Version 1

For this version, the dataset was exports in the format of Yolo v5 Pytorch supported by Yolov5 and then trained on Google Colab.

Yolov5 is based on PyTorch framework rather than fork like Yolov4. Yolov5 also has CSP backbone and PA-NET neck. Yolov5 has advantage in terms of inference time as compared to Yolov4.

5.1.2 Version 2

For this version, the dataset was exports in the format of Yolo Darknet supported by Yolov4 and then trained on Google Colab.

Yolov4 is based on darknet framework which was able to obtain an AP value of 43.5 percent on the COCO dataset along with a inference time of 65 FPS on Tesla V100.

6 Result & Analysis

6.1 Version 1

6.1.1 Model Accuracy Information

The dataset was trained using the modified Yolov5 PyTorch model. The model was trained for 150 epochs that took 3.21 hours to complete.

It resulted in a Precision of 73.6%.

Precision measures the percentage of correct predictions.

$$Precision = \frac{TP}{TP + FP} \tag{1}$$

Recall measures how well a model finds all the positive occurrences.

$$Recall = \frac{TP}{TP + FN} \tag{2}$$

TP = True Positives (Predicted as the occurrence of some object and it was correct)

FP = False Positives (Predicted as the occurrence of some object, but it was incorrect)

6.1.2 Model Performance Metrics

From the graph in figure 3, it is observed that over time precision has gradually increased, but some level of fluctuation can be observed, which suggests that a training model can achieve higher Precision with more epochs.

Figure 3: Model Performance Metrics

Figure 4 shows the mean average precision for the yolov5 model. It was be observed that MAP of average 4.0 was observed out of 5.0 which shows that model's excellent performance.

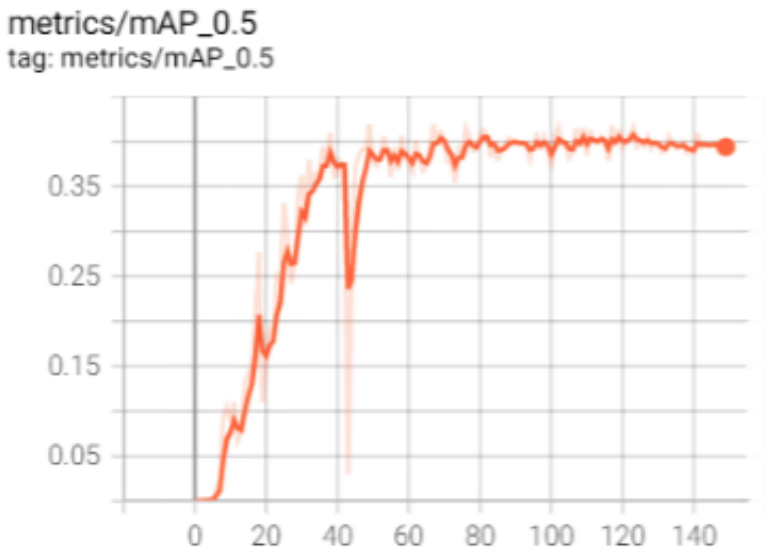

Figure 4: Mean Average Precision

6.2 Version 2

6.2.1 Model Accuracy Information

The dataset was trained on the modified Yolov4 Darknet model. The model was trained for 1000 epochs that took 5.62 hours to complete.

The model resulted in a Precision of 63.4%.

6.2.2 Model Performance Metrics

From the graph in figure 5, it is observed that over time precision has gradually increased but has not become constant yet, which suggests that Precision can be increased by training the model with more epochs.

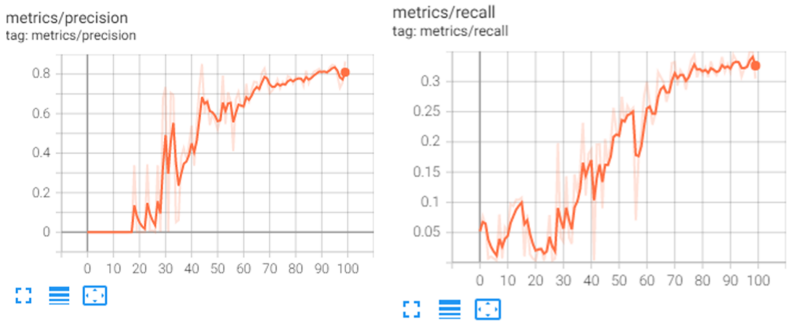

Figure 5: Model Performance Metrics

6.3 Version Comparison

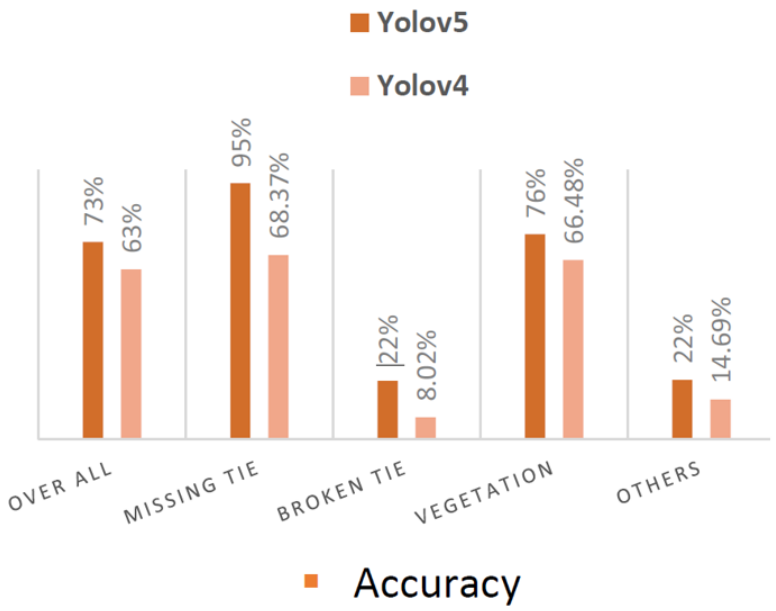

Figure 6: Version Comparison

Figure 6 shows the comparision of accuracy among different categories for both the versions of Yolo.

It can be observed from the figure 6 that contrary to the prior research, Yolov5 shows consistent better performance in each of the category. Where highest accuracy was observed in the categories of vegetation and missing tie with accuracy of 76% and 95% respectively.

7 Example Images

After training the model, images were influx into the model to test its accuracy in detecting vegetation and missing tracks.

7.1 Version 1

7.1.1 Detection of Vegetation

One of the tasks of our model was to detect vegetation on the side and top of railway tracks.

The image in figure 7 shows vegetation detected on top of tracks with confidence score of 0.87 and 0.89. The image in figure 8 shows very small vegetation detected on side of tracks with confidence score of 0.78.

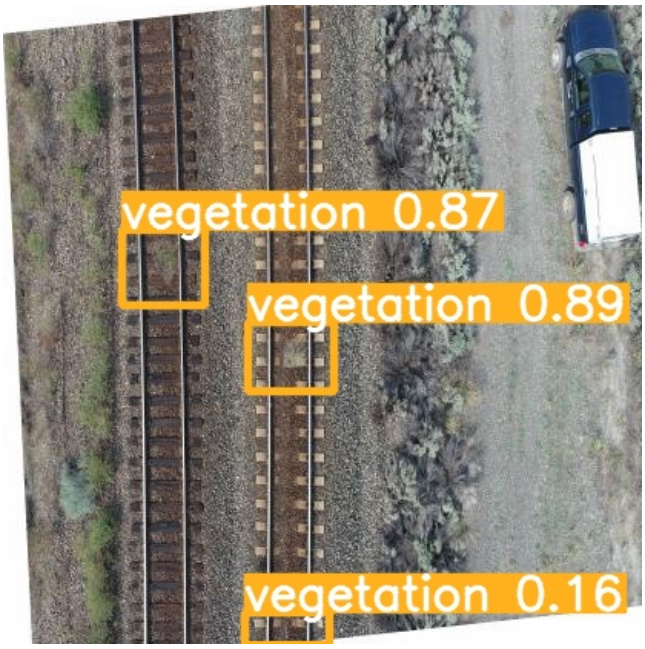

Figure 7: Vegetation on top of railway track

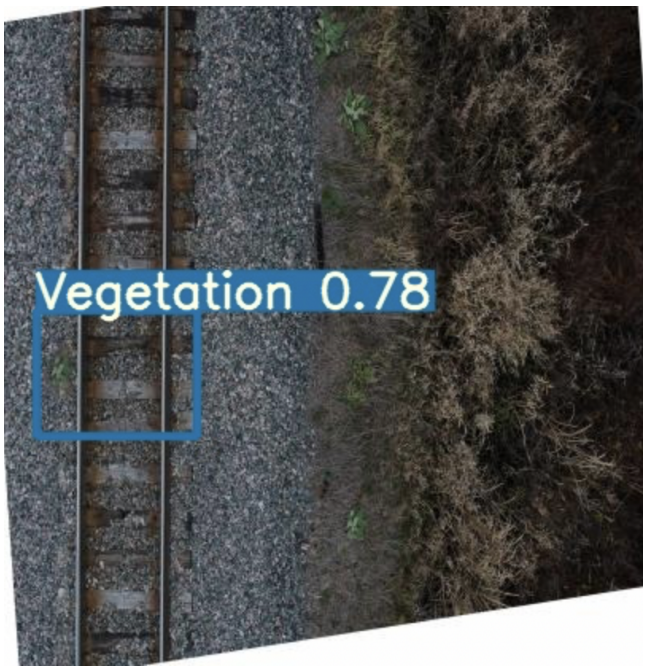

Figure 8: Vegetation on side of railway track

7.1.2 Detection of Missing Ties

Another task of our model was to detect missing ties on railway tracks.

Figure 9 shows model detecting missing tie with confidence score of 0.74.

Figure 10 shows the detection of missing tie as well as broken tie.

Figure 9: Missing Tie

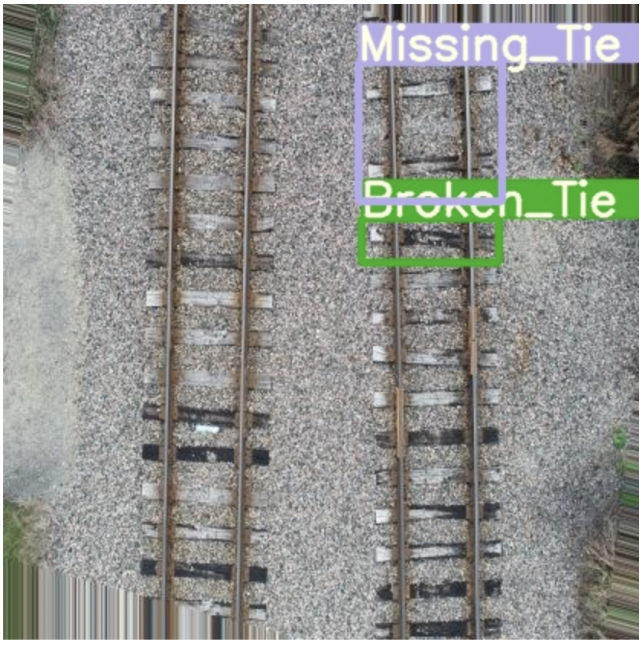

Figure 10: Missing Tie and Broken Tie

7.2 Version 2

7.2.1 Detection of Vegetation

We also tested the second dataset version to detect vegetation on the side and top of railway tracks. The image in figure 11 shows vegetation detected on top of tracks with confidence score of 0.69 and 0.81. The figure 12 shows vegetation detection on side of the track with 0.77 confidence score.

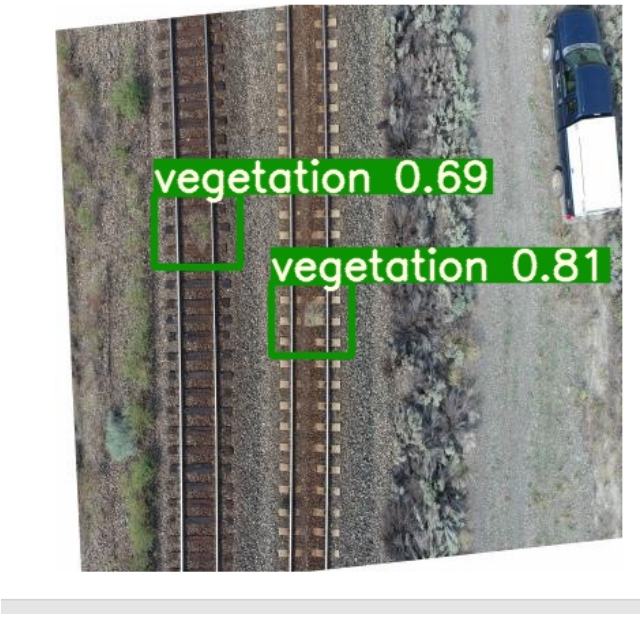

Figure 11: Vegetation on top of railway track

Figure 12: Vegetation on side of railway track

7.2.2 Detection of Missing Ties

The images in figure 13 and 14 show detection of missing ties on railway tracks respectively.

Figure 13: Missing Ties

Figure 14: Missing Ties

8 Conclusion

In a summary, after training yolo algorithm we conclude that the accuracy of various points of interest was directly proportional to same of the factors such as distance from which the images was captured, Augmentation applied and version of the algorithm. Images with broken ties were hard to identify from a greater distance due to low sample size. Also, the algorithm was quickly able to identify the vegetation and missing ties in the images.We can say that it is recommended that the image is captured from a distance where it is easy for the algorithm to identify various points of interest, such as the vegetation, missing ties, and broken ties. Also irrespective of the version of the algorithm, it requires robust training to achieve higher accuracy.

8.1 Key Observations for research questions

- **Image manipulation** techniques such as orientation correction and artificial points of interest makes annotation efficient and accurate.
- **Image Augmentations** have drastic impact on the model performance where colour enhancing augmentations such as saturation and contrast leads to higher precision.
- **Robust training (epoch $>$ 1000) of Yolov5 model** leads to remarkable accuracy among some points of interests such as vegetation and missing tie.

9 Future Work

As this is an ongoing project, future work involves following tasks:

- Retraining the current model based on the recent captured data-set which includes 4.5k images and we expect to see much improvement in the result
- Introducing new category of water pooling and trees in the new dataset as per the requrement of VIA rail
- Comparing performance of different model size in yolov5

10 Acknowledgement

I would like to acknowledge and applaud Dr. Yvonne Coady for her constant motivation and support through this research. I would also like to thank Dr. Aibin for his technical guidance and support. I also appreciate the work of my project partners (Sanyami Shah and Rohan Sharma) for their valuable contribution to this project. I also acknowledge the contribution of my colleagues who constantly evaluated and provided their valuable feedback during each stage of project execution.

References

[1] Yucheng Hu, Zhonghong Ou, Xiangyu Xu, and Meina Song. 2019. *A Crowdsourcing Repeated Annotations System for Visual Object Detection.* Association for Computing Machinery, New York, NY, USA. https://doi.org/10.1145/3387168.3387242

[2] Zicong Jiang, Liquan Zhao, Shuaiyang Li, and Yanfei Jia. 2020. Real-time object detection method based on improved YOLOv4-tiny. *CoRR* abs/2011.04244 (2020). https://arxiv.org/abs/2011.04244

[3] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2016. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. (2016).

[4] Mingxing Tan, Ruoming Pang, and Quoc V. Le. 2020. EfficientDet: Scalable and Efficient Object Detection. (2020).

[5] Wen-Kai Wu, Chien-Yu Chen, and Jiann-Shu Lee. 2021. Embedded YOLO: Faster and Lighter Object Detection. In *Proceedings of the 2021 International Conference on Multimedia Retrieval (ICMR '21)*. Association for Computing Machinery, New York, NY, USA, 560–565. DOI:http://dx.doi.org/10.1145/3460426.3463660

[6] Xin Xu, Shiqin Wang, Zheng Wang, Xiaolong Zhang, and Ruimin Hu. 2021. Exploring Image Enhancement for Salient Object Detection in Low Light Images. *ACM Trans. Multimedia Comput. Commun. Appl.* 17, 1s, Article 8 (mar 2021), 19 pages. DOI:http://dx.doi.org/10.1145/3414839

[7] J. Yu, H. Xie, M. Li, G. Xie, Y. Yu, and C. Chen. 2020. Mobile Centernet for Embedded Deep Learning Object Detection. In *2020 IEEE International Conference on Multimedia Expo Workshops (ICMEW)*. IEEE Computer Society, Los Alamitos, CA, USA, 1–6. DOI:http://dx.doi.org/10.1109/ICMEW46912.2020.9106033

[8] Jiacheng Zhang, Pingyu Wang, Zhicheng Zhao, and Fei Su. 2021. Pruned-YOLO: Learning Efficient Object Detector Using Model Pruning. In *Artificial Neural Networks and Machine Learning – ICANN 2021*, Igor Farkaš, Paolo Masulli, Sebastian Otte, and Stefan Wermter (Eds.). Springer International Publishing, Cham, 34–45.