Real-time Bolt Detection Using Deep Learning

Anastasios Stamoulakatos
National Manufacturing Institute of Scotland
University of Strathclyde

Contents

List of Figures

List of Tables

I. Related Work

Recently, deep learning approaches are taking over inspection processes in different fields with a goal of automating them. Deep learning can yield improved performance as it allows multiple processing layers to learn features by themselves opposite to conventional machine learning approaches which are not able to process the data in their natural form. This rise in deep learning approaches comes as a consequence of the computation demands that these algorithms require being provided in the recent years. Presented here are works in the fields of power line, manufacturing and subsea pipeline inspection. These works highlight the increase of utilising deep learning in several inspection processes. Figure 1 provides a diagram of inspection processes that utilise deep learning. Similar deep learning models and methods can be applied to a wide range of inspection process with the same goal of making the process faster, more accurate and continuous compared to human inspectors.

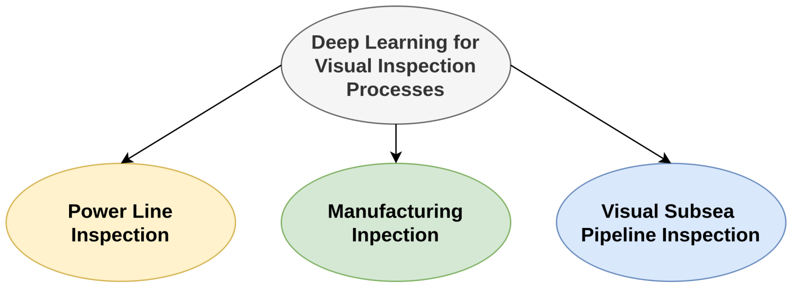


Fig. 1: Inspection Taxonomy

A. Power Line Inspection

One inspection process that deep learning aims to automate is the power line inspection, where deep learning models are used to perform object detection to identify specific features in power line assets and thus make inspection faster and continuous. Liu *et al.* [3] provided an in-depth discussion of deep learning technology in power line inspection, highlighting that after 2015, more than half of the publications about vision-based power line inspection utilise deep learning. Nguyen *et al.* [4] conducted a review on vision-based approaches for power line inspection and the potential role of deep learning and deployed a Single Shot Multibox Detector (SSD) [5] on an Autonomous Underwater Vehicle (AUV) to detect components and faults in power lines [6]. Zhang *et al.* [7] detected electricity poles in Google Street View Imagery using RetinaNet [8] trained with 1,000 annotated images. Jalil *et al.* [9] utilised Faster-RCNN [10] to detect insulators in drone imagery. Miao *et al.* [11] implemented a bespoke SSD with MobileNet [12] as the backbone to detect insulators. Wan *et al.* [13] implemented and compared three custom object detectors for identifying electrical fittings and transmission line defects. Vemula *et al.* [14] deployed and test a Mask R-CNN model [15] on an AUV for detecting power poles, insulators and transformers. Jiang *et al.* [16] analyzed and compared methods for infrared image recognition of power equipment patrol inspection. Han *et al.* [17] used Fast-RCNN [18] for detecting typical defects on equipment of power substations, metere readings, infrared images and humans. They also used a Kalman [19] filter for tracking the detected humans. Most of the aforementioned works use deep learning models that perform object detection to identify specific features in power line assets with the goal of making inspection faster and continuous.

B. Subsea Pipeline Inspection

Given the importance of subsea pipeline inspection, as well as the aforementioned challenges of subsea visual footage, several methods to automate the annotation process using deep learning have been proposed in recent years. The first approach is taken by Foresti *et al.* [20]; they propose a neural network for performing image segmentation of pipeline borders. They argue that this approach better addresses the lack of luminosity as the neural network takes global properties of the images into account. A reasoning based post-processing algorithm was then used to avoid false positives occurring from sea weed and other sea growth. Petraglia *et al.* [21] examine two Neural Network (NN) architectures for classification of four types of events: inner coating exposure, algae, flange and concrete blankets are compared. The first NN architecture utilises two convolutional and three fully connected layers, trained on segmented pipelines from the pre-processed images. The second architecture adopted a Multilayer Perceptron (MLP) with a single hidden layer, trained on features extracted from 3-level Wavelet decomposition. Results led to the conclusion that the convolutional neural network outperforms the MLP, without any need for manual feature extraction. Bharti *et al.* [22] fine-tunes U-Net [23] in a self-supervised setting utilising multi-beam echosounder data for detection and segmentation of subsea pipelines.

C. Manufacturing Inspection

The utilization of deep learning and specifically of CNNs have been widely investigated for machinery fault diagnosis and classification. Yang *et al.* [24] presented a review about the use of deep learning in detecting defects in manufacturing and its challenges. In [25], a deep CNN architecture is designed for defect detection and different hyper-parameters are examined towards the accuracy of the detection results. In [26], a CNN which performs feature extraction directly from the images of steel defects is compared with traditional machine learning (a combination of manual feature extraction and a multi-layer perceptron and support vector machine) and shows lower error rates. Another CNN is explored in [27] to automatically inspect dirties, scratches, burrs, and wears on surface parts and the results show that it works properly with different types of defects on textured or non-textured surfaces. A generic approach based on CNN is proposed in [28] to extract patch feature and predict defect area via thresholding and segmenting. CNNs have been also used in other applications of defect diagnosis such as bearing [29]–[31], gearbox [32] and rotors [33]. Liu *et al.* [34] utilised pix2pix [35] conditional Generative Adversarial Network (GAN) [36] to generate more samples and tackle the class imbalance that exists in industrial processes and DenseNet [37] to detect defects in these samples. In [38], [39] state-of-the-art deep learning methods are presented for inspecting surface defects in industrial products. In addition, they implemented a semi-supervised [40] framework for identifying manufacturing defects while they introduced a loss component and made extensive use of mix augmentations [41].

II. Methodology

The goal of this work is to perform real time detection and counting of bolts on boxes using a computer vision deep learning model. A sample photo of the box is seen in Figure 2. Upon success, this framework will make manufacturing process faster and cheaper by having a surveillance system active twenty four hours per day. Figure 3 illustrates the project workflow to create a real time detector. Firstly, to utilise a deep learning model a dataset has to be collected to train the model on. Then the chosen model is trained on this dataset and its performance is measured through predictions and metrics. Once the performance is at satisfactory levels the real time detection is implemented using the appropriate hardware. The whole process can be iterative as training the model again with more and diverse data will create more robust models, that can adapt to different backgrounds, lighting conditions and distances of the box from the camera.

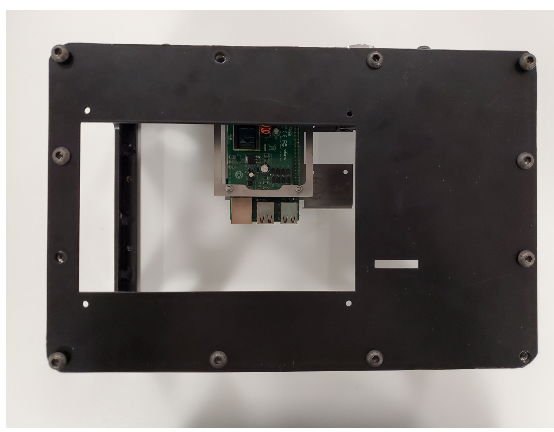


Fig. 2: Example data

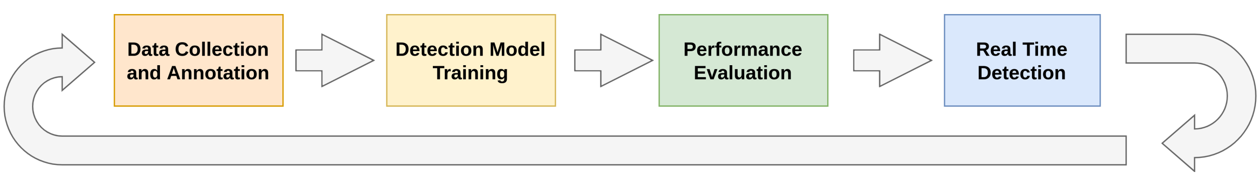
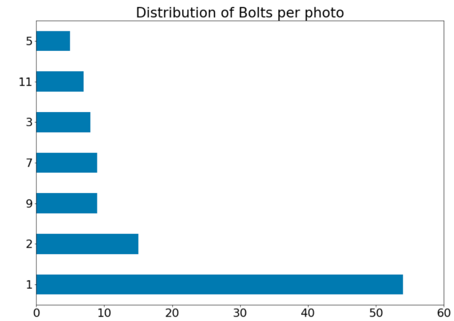


Fig. 3: Project Workflow

A. Data Collection and Annotation

The first step in creating a deep learning model is to create the dataset to train the model on. This step requires manual intervention. Photos of the boxes with different number of bolts are taken; 112 in total. For creating the annotation bounding

(a) Annotation with CVAT



(b) Distribution of bolts per box

boxes CVAT (https://github.com/openvinotoolkit/cvat) is used. It is an open source, online, interactive image annotation tool for computer vision. A screenshot of the annotation is process is seen in Figure 4a. The annotations can be exported in different formats (json, xml, txt) depending on what the architecture requires. For example, to train EfficientDet [1] *xml* annotation files are used where a bounding box is described by the coordinates of its top-left (xmin, ymin) corner and its bottom-right (xmax, ymax) corner, while YOLOv5 [2] requires *txt* data with bounding boxes in the format of xcenter, ycenter, width, height. The maximum number of bolts a box can have is 12. Figure 4b provides the distribution of bolts per photo. It is evident that most of the photos taken (over 50) contain one bolt while the other frequencies are similar.

B. *EfficientDet*

Object detection is an advanced form of image classification where a neural network predicts objects in an image and points them out in the form of bounding boxes. Object detection thus refers to the detection and localization of objects in an image that belong to a predefined set of classes. In computer vision applications the pixels of an image are translated into features using neural networks. Major progress has been made in the field of computer vision by using convolutional neural networks (CNN) to create learnable features from an image. Convolutional neural networks mix and pool image features at different levels of granularity, allowing the model a choice of possible combinations to focus on when learning the image detection task at hand.

EfficientDet [1] is a type of object detection model, which utilizes several optimization and backbone tweaks, such as the use of a BiFPN. BiFPN is a bidirectional feature pyramid network which incorporates the multi-level feature fusion and enables information to flow in both the top-down and bottom-up directions, while using regular and efficient connections, as seen in Figure 5. This model is chosen as it provides high performance in object detection tasks. However, models with similar performance can also be used for this application, provided they have a short inference time that will provide real time detection.

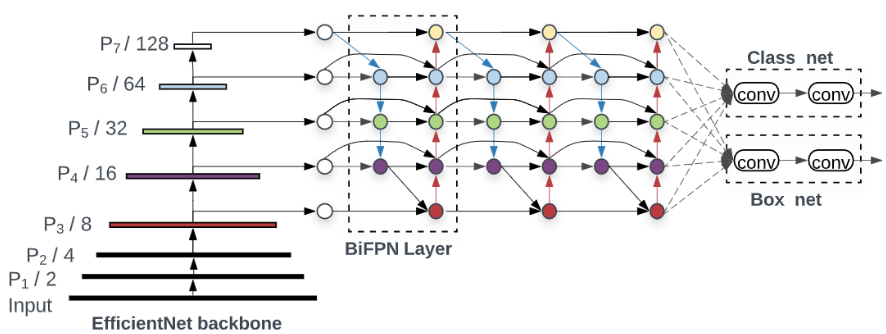


Fig. 5: EfficientDet [1]

1) Training an Object Detector: The training process is an optimization problem. The parameters are learned during model training through sample data and optimising (minimising) the output error [42]. Figure 9a provides a logical diagram of the training process for a neural network in a supervised setting where input data $\mathbf{x}$ and their corresponding labels $\mathbf{y}$ are available. In supervised learning, the loss function calculates the error between the network output $\hat{\mathbf{y}}$ and the ideal solution $\mathbf{y}$ for the current set of parameters θ. The goal is to minimize a loss function denoted by $\mathcal{L}(\theta)$. Loss is a non-negative value and as it decreases, the performance of the model improves. One important aspect in supervised learning is the choice of an appropriate loss function for the task. For example, it is common to use Mean Squared Error losses for regression tasks, and Cross Entropy losses for classification tasks. Here, $\mathbf{x}$ is the images, and $\mathbf{y}$ are the labels (bolt, background) and the bounding boxes (coordinates of boxes in the image). Identifying a label is a classification problem, while identifying a bounding box is a regression problem.

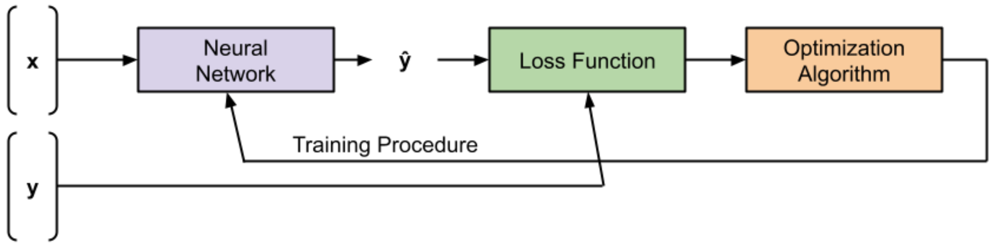


Fig. 6: Training in Supervised Learning

2) Augmentations: Augmentations are techniques that modify that training dataset and thus improve the model's generalization capability. It is widely used in computer vision tasks to increase the size of the dataset, to diversify the samples and thus make the model more robust and improve its generalization capabilities [43]. Augmentation can be made offline (creating more samples before the training of a model) or online during the training process (selected here). In this work, the augmentations used are horizontal and vertical flipping, blurring, rotations with 15 degrees maximum angle, hue saturation and random brightness and contrast changes. Augmentations can address the changes in the lighting conditions of a workplace and the angle a photo is taken. Examples of augmentations used are shown in Figures 7, 8.

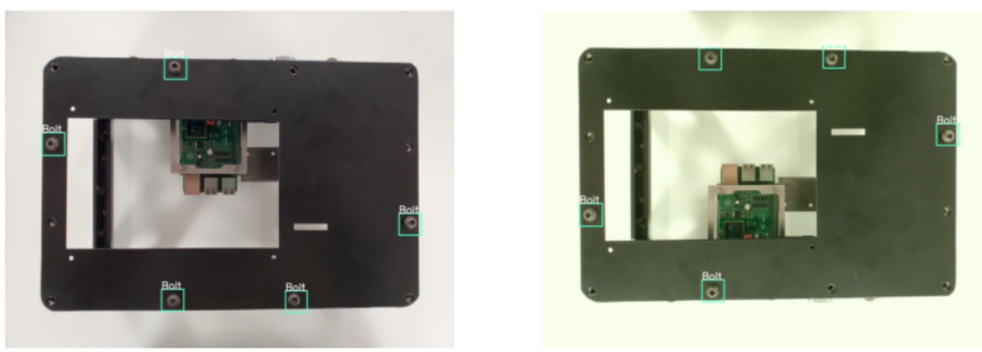


Fig. 7: Augmentations

3) Training and Inference on Test Set: The goal of the training is to minimize the error functions for both the training and validation losses as seen in Figure 9. The model is trained for 200 epochs with AdamW [44] optimizer and a learning rate of 0.001. Images are resized to $1024x1024$ to tackle GPU memory constraints. The prediction threshold is set to 0.2. The model with parameters that obtain the lowest validation loss is saved as the best model. The training losses are logged using Tensorboard [45].

After training model, it is essential to know how it performs in unseen data and consequently whether its predictions can be trusted. To evaluate the performance of a model in supervised learning it is common to split the original dataset in three sets (60%, 20%, 20%) and calculate the performance metrics on samples that are not used during the training phase. The training set (60%) used to build predictive models. The validation set is used to assess the performance of the model built in the training phase. It provides a test platform for fine-tuning a model's hyperparameters and selecting the best performing model. Finally the test set, or unseen data, is used to assess the likely future performance of a model. If a model fits to the training set much better than it fits the test set, overfitting is probably the cause. The purpose of holdout test evaluation is to test a model on different data than it was trained on which provides an unbiased estimate of learning performance. The holdout approach is useful because of its speed, simplicity, and flexibility.

Mean Average Precision (mAP) is the most popular metric used to evaluate object detection models. EfficientDet achieved a mAP of 0.766 for the validation set and a mAP of 0.720. Both results are similarly high which means that the model can

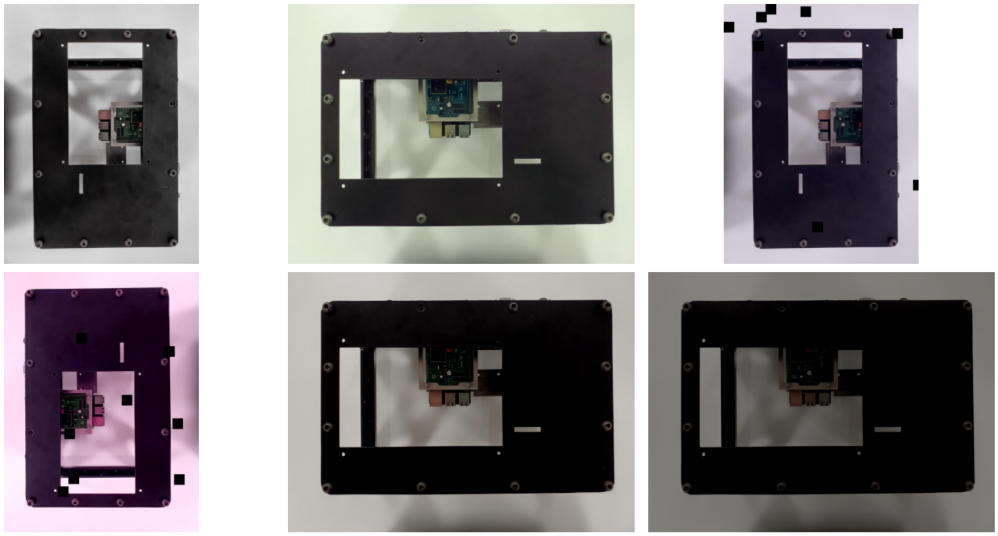
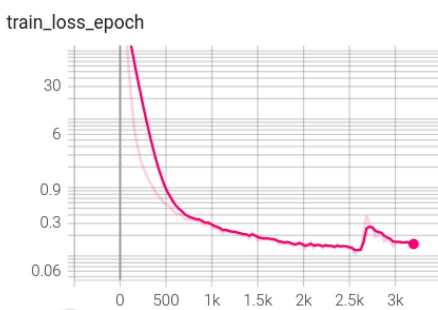
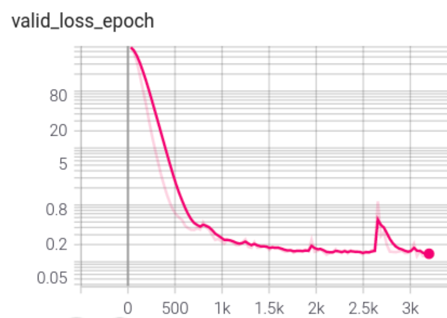


Fig. 8: More augmentations

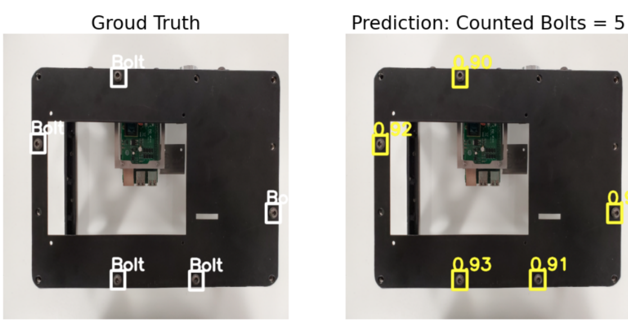


(a) Training loss

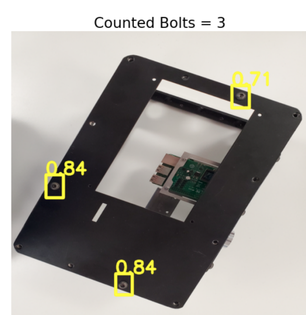


(b) Validation loss

Fig. 9: Measuring the losses during training



(a) Inference on the test set



(b) inference on new image

Fig. 10: Inference

generalize well to unseen test images. In Figure 10a predicted bolts on one image of the test set are illustrated along with their confidence scores and their ground truth bounding boxes. Figure 10b shows a prediction on a new photo, provided using a phone, that has a different angle in reference to the box. The model performs equally well, on predicting the positions of the bolts, but with a lower confidence score which is expected as this image differs from the ones used for training.

C. YOLOv5

YOLOv5 is another state-of-the-art object detection model that provides similar performance with EfficientDet in benchmark datasets, but is faster beating other real-time object detection algorithms by a large margin as seen in Figure 11. YOLOv5 [2] is an open-source project that consists of a family of object detection models and detection methods based on the YOLO model pre-trained on the COCO [46] dataset. It is maintained by Ultralytics and represents the organization's open-source research into the future of Computer Vision works. Following the documentation of ZED, YOLOv5 is selected as it is compatible with the ZED camera and the Edge AI Gateway device. They provide a script for running object detection inference (https://github.com/stereolabs/zed-examples).

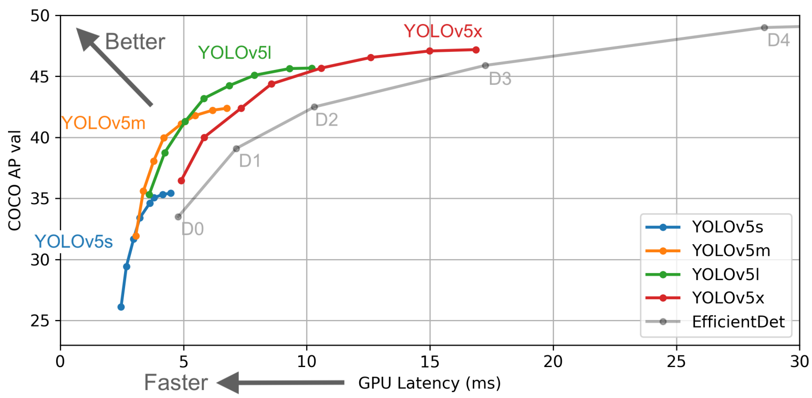


Fig. 11: YOLOv5 performance [2]

1) Training and Inference on Test Set: A similar training procedure, as in EfficientDet, is followed for YOLOv5. The model is trained for 200 epochs with a learning rate of 0.01 and SGD [47] optimizer. Cycling scheduler policy is applied [48], as well as augmentations of flipping, translation, scaling and hue saturation. Similarly with EfficientDet the model that provides the lowest validation loss is kept at the end of the training. Figure 12a shows how the training and validation losses and metrics progress during training, while Figure 12b is an example of YOLOv5 inference on a test image. YOLOv5 achieved a mAP of 0.758 for the validation set and a mAP of 0.762 for the test set. The performance is similar with EfficientDet for the current dataset, but the inference time is shorter.

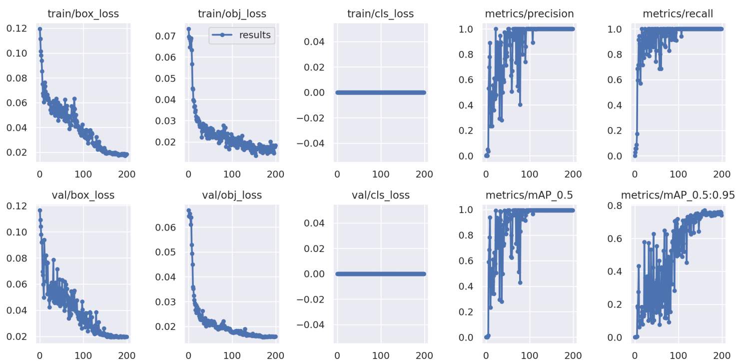
D. Real time detection

The training of the model is done using a server equipped with a NVIDIA GeForce RTX 3090 GPU with 24GB of RAM and the performance of the model is evaluated using the test set. To perform real time bolt detection, the hardware of Figure 13a is used. The ZED camera provides live video feed that can reach 100 FPS and the Stereolabs Edge AI Gateway is a compact gateway powered by NVIDIA Jetson that allow model deployment. The Edge device works as an Ubuntu machine as seen in Figure 13b. By transferring the model to the Edge real time detection is performed as seen in Figure 14 where the top right bolt of the box is detected.

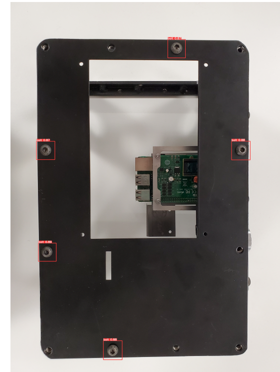
1) Limitations and Next Steps: Performing real time bolt detection with frames provided by the ZED camera presented some limitations. As seen in Figure 14 only the top left bolt is detected out of four bolts present on the box. The reason of this inadequate performance is that the model is trained with photos taken from a phone (Figure 12b) where the background of the box is clear. However, on inference, frames provided by the ZED camera feed contain a different background, lighting condityoloions, distance of the box from the camera and different input resolution. To address that, a new dataset will be created by using the ZED camera to collect data instead of a phone camera.

III. Summary

This work provides the methodology of a complete deep learning pipeline of training an object detection model and using the appropriate hardware to deploy the model and perform real time detection. Two deep learning models are examined that

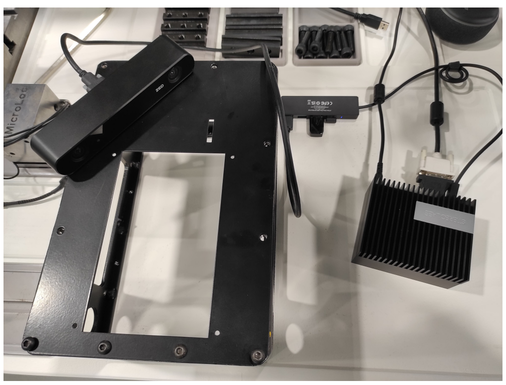


(a) YOLOv5 losses and metrics

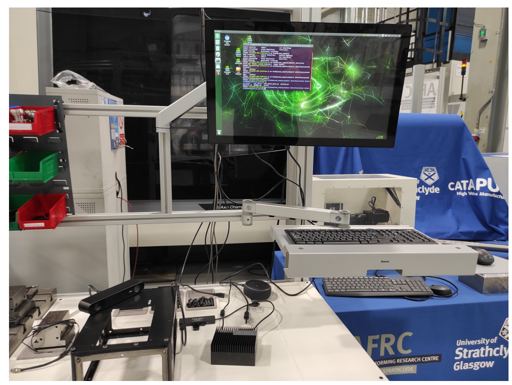


(b) Predictions on a test image

Fig. 12: Training and Inference



(a) Hardware for real time detection



(b) Edge device as Ubuntu machine

Fig. 13: Hardware for real time detection

provide similar performance when evaluated on the same test set. YOLOv5 is chosen as it provides faster inference time and it is also compatible with the hardware specifications of the ZED camera and the Edge AI Gateway. Following work will be focused on the necessary steps to improve the performance of the real time detection, by extracting and training the model with a new dataset with images that contain similar conditions to the real time environment.

References

[1] M. Tan, R. Pang, and Q. V. Le, "Efficientdet: Scalable and efficient object detection," 2020.

[2] "ultralytics/yolov5: v3.1 - Bug Fixes and Performance Improvements," Oct. 2020. [Online]. Available: https://doi.org/10.5281/zenodo.4154370

[3] X. Liu, X. Miao, H. Jiang, and J. Chen, "Data analysis in visual power line inspection: An in-depth review of deep learning for component detection and fault diagnosis," *Annual Reviews in Control*, vol. 50, p. 253–277, 2020. [Online]. Available: http://dx.doi.org/10.1016/j.arcontrol.2020.09.002

[4] V. N. Nguyen, R. Jenssen, and D. Roverso, "Automatic autonomous vision-based power line inspection: A review of current status and the potential role of deep learning," *International Journal of Electrical Power & Energy Systems*, vol. 99, pp. 107 – 120, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0142061517324444

[5] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," *Lecture Notes in Computer Science*, p. 21–37, 2016. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-46448-0_2

[6] V. N. Nguyen, R. Jenssen, and D. Roverso, "Intelligent Monitoring and Inspection of Power Line Components Powered by UAVs and Deep Learning," *IEEE Power and Energy Technology Systems Journal*, vol. 6, no. 1, pp. 11–21, 2019.

[7] W. Zhang, C. Witharana, W. Li, C. Zhang, X. Li, and J. Parent, "Using deep learning to identify utility poles with crossarms and estimate their locations from google street view images," *Sensors (Switzerland)*, 2018.

[8] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," 2018.

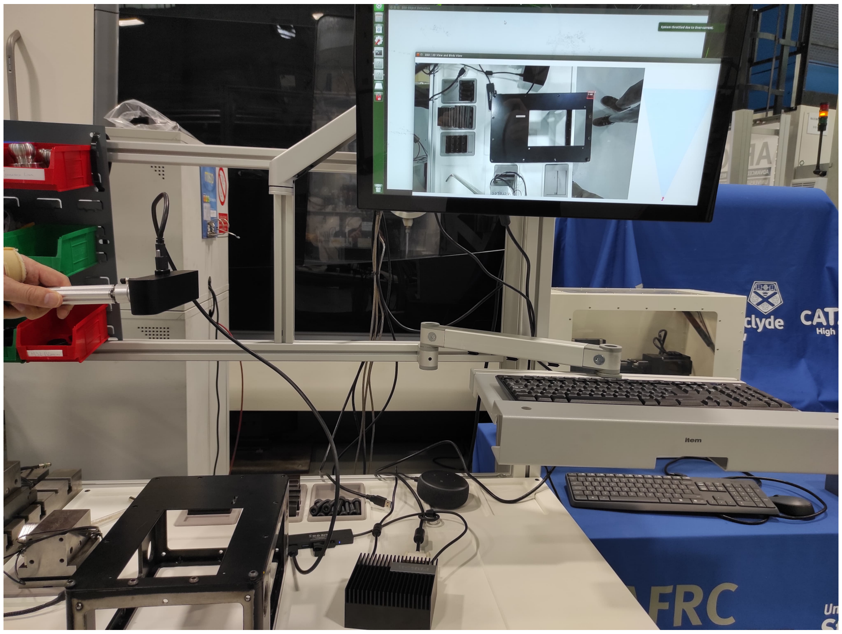


Fig. 14: Real time detection

[9] B. Jalil, G. R. Leone, M. Martinelli, D. Moroni, M. A. Pascali, and A. Berton, "Fault Detection in Power Equipment via an Unmanned Aerial System Using Multi Modal Data," *Sensors*, 2019.

[10] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN," 2017.

[11] X. Miao, X. Liu, J. Chen, S. Zhuang, J. Fan, and H. Jiang, "Insulator detection in aerial images for transmission line inspection using single shot multibox detector," *IEEE Access*, 2019.

[12] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," 2017.

[13] N. Wan, X. Tang, S. Liu, J. Chen, K. Guo, L. Li, and S. Liu, "Transmission Line Image Object Detection Method Considering Fine-Grained Contexts," *Proceedings of 2020 IEEE 4th Information Technology, Networking, Electronic and Automation Control Conference, ITNEC 2020*, no. Itnec, pp. 499–502, 2020.

[14] S. Vemula and M. Frye, "Mask R-CNN powerline detector: A deep learning approach with applications to a UAV," *AIAA/IEEE Digital Avionics Systems Conference - Proceedings*, vol. 2020-Octob, pp. 1–6, 2020.

[15] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," 2018.

[16] A. Jiang, N. Yan, B. Shen, C. Gu, H. Zhu, and H. Huang, "Research on Infrared Image Recognition Method of Power Equipment Based on Deep Learning," *7th IEEE International Conference on High Voltage Engineering and Application, ICHVE 2020 - Proceedings*, pp. 12–15, 2020.

[17] R. Han, L. Liu, S. Liu, P. Jiang, Y. Han, and Z. Yang, "Research and application of substation intelligent inspection technology based on multi spectral image recognition," *7th IEEE International Conference on High Voltage Engineering and Application, ICHVE 2020 - Proceedings*, pp. 12–15, 2020.

[18] R. Girshick, "Fast r-cnn," in *2015 IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 1440–1448.

[19] G. Welch and G. Bishop, "An introduction to the kalman filter," University of North Carolina, USA, Tech. Rep., 1995.

[20] G. L. Foresti and S. Gentili, "Vision based system for object detection in underwater images," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 14, no. 2, pp. 167–188, 2000.

[21] F. R. Petraglia, R. Campos, J. G. R. C. Gomes, and M. R. Petraglia, "Pipeline tracking and event classification for an automatic inspection vision system," in *2017 IEEE International Symposium on Circuits and Systems (ISCAS)*. Baltimore, MD, USA: IEEE, May 2017, pp. 1–4. [Online]. Available: http://ieeexplore.ieee.org/document/8050761/

[22] V. Bharti, D. Lane, and S. Wang, "Learning to Detect Subsea Pipelines with Deep Segmentation Network and Self-Supervision," *2020 Global Oceans 2020: Singapore - U.S. Gulf Coast*, 2020.

[23] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," 2015.

[24] J. Yang, S. Li, Z. Wang, H. Dong, J. Wang, and S. Tang, "Using deep learning to detect defects in manufacturing: A comprehensive survey and current challenges," *Materials*, vol. 13, no. 24, pp. 1–23, 2020.

[25] D. Weimer, B. Scholz-Reiter, and M. Shpitalni, "Design of deep convolutional neural network architectures for automated feature extraction in industrial inspection," *CIRP Annals*, vol. 65, no. 1, pp. 417–420, 2016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0007850616300725

[26] M. J., M. U., C. D., S. J., and F. G., "Steel defect classification with max-pooling convolutional neural networks [c] neural networks (ijcnn)," *The 2012 International Joint Conference on Neural Networks (IJCNN)*, p. 1 – 6, 2012, cited by: 5. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85091730826&partnerID=40&md5=ca7435e1319fb3ee08ddd58acaf5452b

[27] R. Ren, T. Hung, and K. C. Tan, "A generic deep-learning-based approach for automated surface inspection," *IEEE Transactions on Cybernetics*, vol. 48, no. 3, pp. 929–940, 2018.

[28] J. K. Park, B. K. Kwon, J. H. Park, and D. J. Kang, "Machine learning-based imaging system for surface defect inspection," *International Journal of Precision Engineering and Manufacturing - Green Technology*, vol. 3, no. 3, pp. 303–310, 2016.

[29] X. Guo, L. Chen, and C. Shen, "Hierarchical adaptive deep convolution neural network and its application to bearing fault diagnosis," *Measurement*, vol. 93, pp. 490–502, 2016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0263224116304249

[30] C. Lu, Z. Wang, and B. Zhou, "Intelligent fault diagnosis of rolling bearing using hierarchical convolutional network based health state classification," *Advanced Engineering Informatics*, vol. 32, pp. 139–151, 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1474034616301148

[31] V. D., D. E., M. V., M. M., and F. A., "Deep learning enabled fault diagnosis using time-frequency image analysis of rolling element bearings," *Shock Vib*, vol. 2017, p. 1 – 29, 2017, cited by: 4. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85053418750&partnerID=40&md5=8dd563aed3b2f2be481b5f74f4a995f3

[32] P. Wang, Ananya, R. Yan, and R. X. Gao, "Virtualization and deep recognition for system fault classification," *Journal of Manufacturing Systems*, vol. 44, pp. 310–316, 2017, special Issue on Latest advancements in manufacturing systems at NAMRC 45. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0278612517300511

[33] O. Janssens, V. Slavkovikj, B. Vervisch, K. Stockman, M. Loccufier, S. Verstockt, R. Van de Walle, and S. Van Hoecke, "Convolutional neural network based fault detection for rotating machinery," *Journal of Sound and Vibration*, vol. 377, pp. 331–345, 2016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0022460X16301638

[34] Y. W. Lu, K. L. Liu, and C. Y. Hsu, "Conditional Generative Adversarial Network for Defect Classification with Class Imbalance," *Proceedings - 2019 IEEE International Conference on Smart Manufacturing, Industrial and Logistics Engineering, SMILE 2019*, pp. 146–149, 2019.

[35] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," 2018.

[36] M. Mirza and S. Osindero, "Conditional generative adversarial nets," 2014.

[37] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," 2018.

[38] X. Zheng, H. Wang, J. Chen, Y. Kong, and S. Zheng, "A Generic Semi-Supervised Deep Learning-Based Approach for Automated Surface Inspection," *IEEE Access*, vol. 8, pp. 114 088–114 099, 2020.

[39] X. Zheng, S. Zheng, Y. Kong, and J. Chen, "Recent advances in surface defect inspection of industrial products using deep learning techniques," *International Journal of Advanced Manufacturing Technology*, vol. 113, no. 1-2, pp. 35–58, 2021.

[40] X. J. Zhu, "Semi-supervised learning literature survey," University of Wisconsin-Madison Department of Computer Sciences, Tech. Rep., 2005.

[41] D. Hendrycks*, N. Mu*, E. D. Cubuk, B. Zoph, J. Gilmer, and B. Lakshminarayanan, "Augmix: A simple method to improve robustness and uncertainty under data shift," in *International Conference on Learning Representations*, 2020. [Online]. Available: https://openreview.net/forum?id=S1gmrxHFvB

[42] P. Sharma and A. Singh, "Era of deep neural networks: A review," in *2017 8th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*. Delhi: IEEE, Jul. 2017, pp. 1–5. [Online]. Available: http://ieeexplore.ieee.org/document/8203938/

[43] C. Shorten and T. M. Khoshgoftaar, "A survey on Image Data Augmentation for Deep Learning," *Journal of Big Data*, vol. 6, no. 1, 2019. [Online]. Available: https://doi.org/10.1186/s40537-019-0197-0

[44] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," 2019.

[45] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: https://www.tensorflow.org/

[46] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. Dollár, "Microsoft coco: Common objects in context," 2015.

[47] S. Ruder, "An overview of gradient descent optimization algorithms," *arXiv preprint arXiv:1609.04747*, 2016.

[48] L. N. Smith, "Cyclical learning rates for training neural networks," 2017.