APPLICATION OF MACHINE LEARNING MODELS FOR POWDER BED DEFECT ANALYSIS

F. Du Rand[1*], P.J.M. Van Tonder[2] & A.F. van der Merwe[3]

[1]Department of Computer Systems and Process Control
Vaal University of Technology, South Africa
francoisdu@vut.ac.za

[2]Technology Transfer and Innovation,
Vaal University of Technology, South Africa
malanvt@vut.ac.za

[3]Department of Industrial Engineering,
Stellenbosch University, South Africa
andrevdm@sun.ac.za

ABSTRACT

The monitoring of the quality of parts that are manufactured using additive manufacturing has become vital, especially with the integration of these technologies alongside traditional manufacturing processes. In-situ monitoring methods have been devised to monitor the quality of parts throughout the manufacturing process. However, in-situ monitoring of builds during the manufacturing process still lacks effective autonomous monitoring strategies. In this paper a computer vision-based machine learning re-coater defect monitoring system has been developed that can classify defects instead of simply detecting its presence on the powder-bed surface.

[1] The author is enrolled for an PhD (Industrial) degree in the Department of Industrial Engineering, Stellenbosch University

1. INTRODUCTION

When manufacturing parts for industries such as aerospace or medical, the parts must be manufactured according to very specific industry quality standards [1]. This requires additional systems that can be used to monitor the manufacturing lifecycle of a part to ensure the traceability and conformance for each step of the manufacturing process. In order to assist with the monitoring of the manufacturing quality of powder bed fusion process, specific techniques have been developed that can be used to monitor each step of the manufacturing process. Some of these techniques include melt pool analysis [2], re-coater monitoring [3], post scan analysis [4] and spatter analysis [5]. The monitoring of the re-coating process has been examined by several researchers, specifically looking at the detection of recoating defects [6] and in defects that occur during the sintering and melting process [3]. Most of these methods make use of computer vision cameras for the imaging of the powder bed surface before and after re-coating. These images are then processed using a variety of image processing techniques to detect defects on the powder bed. Studies performed by Scime and Beuth, from the Carnegie Mellon University [7][8], demonstrated the feasibility of using machine learning for the classification of defects that occurred during the re-coating and sintering process. The classification of defects that occur during the re-coating and sintering is critical for the development of a closed-loop feedback system for additive manufacturing technologies [8]. Both of these studies achieved defect classification by training a machine learning (ML) model with a set of training images, and then used this model to isolate and detect defects on the images captured during an actual build of the powder bed surface. Scime and Beuth have proved the effectiveness of applying a ML model to detect and classify defects on the powder bed surface. However, the method used by the researchers could be streamlined by using more optimised methods that could reduce the amount of overhead processing required and hopefully speed up the detection process and increase the defect detection accuracy.

2. BACKGROUND

In the field of Computer vision, image processing and ML have been used hand in hand for a variety of scenarios, such as face, object, pattern and character recognition and object tracking on images or videos [9]. The purpose of combining ML with computer vision is an attempt to have a computer autonomously perform the same tasks that is naturally performed by the human eyes and brain. However, to train a computer to have the same level of object or scene recognition as the human brain requires a lot of computing power and specialised mathematical models. Over recent years several types of mathematical models have been developed for specific applications and have produced very good results [10]. ML and computer vision have also been used for various applications in the field of additive manufacturing [11]. A number of these studies have used computer vision and ML models for the monitoring of different types of AM technologies such as fused deposition modelling (FDM) processes [11]. Some of the other studies have also looked at the application of computer vision and ML on powder bed-based AM technologies for things such as melt pool analysis [12]. As discussed in the previous section, two studies conducted by Scime and Beuth were identified that have used machine learning models for identifying and classifying re-coater defects on powder bed-based AM machines [7] [8]. These studies made use of a ML technique called image classification. Image classification that determines what class the premise of the image belongs to e.g. An image containing a dog will be classified as dog.

Using this background information, Scime and Beuth trained an ML model using images of different types (classes) of defects. Then, in order to detect and classify defects that may be present on the image, the entire image was split into patches. The image that is to be analysed were sliced into 3 different size patches of 25x25 pixels, 100x100 pixels and 900x900 pixels. The authors made use of 3 different sizes of image patches because some of the defects were bigger than others and an image patch can only contain one type of defect. This way all the different types of defects could be accommodated. These patches were then analysed by the model. Once each patch has been analysed, the model outputs the level of confidence that it has for each defect type that it may have detected in the image patch.

The image patch will be classified as containing the type of defect that it had the highest level of confidence for. However, since most of the powder bed surface will not be containing defects, images of the smooth powder bed had to be added as a no-defect class or type to prevent false positives. This technique proved very effective in the detection and classification of defects. Unfortunately, there are a lot of image processing overhead operations that means that the technique takes between 4-7 seconds to classify the image, let alone the splitting up of the images into patches and other processes. This is close to the amount of time it takes a machine to complete a re-coating cycle. Although this is still acceptable, it would be much more advantageous for a real-time feedback loop if the images could be processed before the re-coating cycle is complete or the sintering process is completed. An alternative ML technique exists, called object detection that has the potential to identify multiple objects in a single image, hopefully within a shorter timeframe. This technique does not require the image to be split into smaller blocks as the entire image can be processed at once. This eliminates a lot of the image pre-processing overheads that is required by the previous method for this specific application. For this research paper, the effectiveness of using an object detection model will be evaluated to determine if it can be used for this application. A model will be selected, trained and evaluated to determine its accuracy and speed at which it can analyse images. Although image classification models as used by Scime and Beuth differ when compared to object detection models, they do produce a measurable result which consists of two factors, namely speed and accuracy. These to factors will be compared then to determine whether the object detection model has a better speed or accuracy compared to the image classification model when used in an AM application.

2.1 Selection of object detection ML Models

In order to select the appropriate object detection model, two key points must be considered, namely the accuracy of the model and the speed of the model. The accuracy of a ML model for object detection is determined by the mean average precision or mAP. The mAP of an object detection ML model is governed by the precision and recall numbers [13]. The precision of a model is determined by the model's ability to predict true positives compared to false positives. The recall of the model is determined by how many true positives the model can predict compared to false negatives. In order to provide literary background, Equation 1,2 and 3 is used to calculate the mAP metric is included as follows:

$$precision = \left(\frac{TP}{TP + FP}\right) \quad \textbf{(1)}$$

$$recall = \left(\frac{TP}{TP + FN}\right) \quad \textbf{(2)}$$

$$mAP = 2 \times \left(\frac{precision \times recall}{precision + recall}\right) \quad \textbf{(3)}$$

Where:
TP = True Positive
TN = True Negative
FP = False Positive
FN = False Negative

The values for TP, TN, FP and FN can only be determined once the model has been trained and evaluated. These values are then used to calculate the precision, recall and mAP values.

It can be inferred that the higher mAP metric a model has, the more precise a specific type of model is. Since it is anticipated that some of the defects that must be detected by the model will be very fine defects, an accurate model will be required. Also based on the limited amount of time that is available between recoating/fusion cycles (7-10 seconds depending on machine type), a model with a moderate speed performance will be required. There are 2 commonly used object detection ML models, namely Single Shot Detector (SSD) and Regionbased Convolutional Neural Network (RCNN) models. It is worth noting that literature has revealed that "Faster-RCNN" models are more accurate when it comes to detecting smaller objects on images and is better suited to applications requiring high levels of accuracy whereas SSD models are faster and more often considered for real-time applications but at a lower accuracy [14].

The models that will be considered for this application are called pre-trained models, thus the mAP metric is calculated based on the dataset that was used to pre-train the model. In this case, the models were pre-trained using the common objects in context (COCO) dataset. The COCO dataset contains various type of images that contains things like vehicles, persons, animals and more and is often used to train or evaluate ML models. Table 1 shows the mAP and speed values for the more commonly used COCO pretrained models available as part of the Tensorflow software platform. It must also be noted that the input image size that these models where trained with was 224 x 224 pixels.

Table 1 ML models for object detection [15]

Model Number	Model Name	mAP metric	Speed (ms)
1	Ssd_mobilenet_v1_coco	21	30
2	ssd_mobilenet_v1_0.75_depth_coco	18	26
3	ssd_mobilenet_v1_quantized_coco	18	29
4	faster_rcnn_inception_v2_coco	28	58
5	faster_rcnn_resnet101_coco	32	92
6	faster_rcnn_inception_resnet_v2_atrous_coco	37	620
7	faster_rcnn_nas	43	1833

When considering the models contained in Table 1, it can be seen that the models that are contained in the Tensorflow model zoo that have the highest accuracy are model no.6 and 7. However, it is also clear that these models are not the fastest. Unfortunately, it is also clear that to have a faster model, the trade-off will be a lower accuracy as can be seen with the SSD models. When comparing the Faster-RCNN models, the most accurate model has a speed of 1833ms or 1.8 seconds. Since these models were pre-trained with images that only had a resolution of 224 x 224 pixels, it means that when processing higher resolution images, the model will perform slower than what is predicted by the model zoo. The images that will be analysed cannot be resized to a smaller resolution because this will result in the loss of fine detail on the image. Since the images to be analysed are 3264 x 2448 pixels in size which is over 100 times bigger than the normal input to the model, it can be estimated that the model will take 100 times slower to process the larger image. However, due to variations in the content contained in the images, it is extremely difficult to theoretically determine the exact performance of model given these bigger images. It can thus only be assumed that the model will perform approximately 100 times slower than predicted. Since it takes a powder bed-based AM machine between 3-7 seconds to complete a re-coating cycle depending on the manufacturer, a model will have to be selected that would have a predicted performance within this range or less. The first 2 faster-RCNN models have mAP values that are significantly higher than the SSD models, but not as high as model no.7 . However, in this case a reduction in accuracy will be considered acceptable as the speed of the model increases significantly. For this cause model no.4 will be tested in this application as it has a reasonable accuracy as well as n relatively high operating speed. This research paper will not be focusing on the technical and mathematical aspects of the different machine learning models, but more on the suitability of the selected ML model to this AM monitoring application.

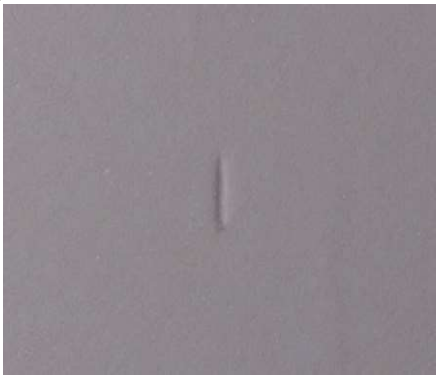
3. METHODOLOGY

3.1 Preparation of the training images

In order to successfully train machine learning model, a sufficiently sized recoater defect dataset is required to ensure that the model achieves an adequate level of learning to detect defects with a very high degree of accuracy. If the training dataset does not contain enough training images of each type of defect, the performance of the model will be very limited [16]. Additionally, in some cases if the dataset is too small it may lead to a condition called overfitting. Overfitting occurs when the model starts to identify patterns in the data that does not exist and as a result, perform poorly when predicting results from new data [17]. For this study, 3 different types of commonly seen recoater defects were selected [18], namely: debris Figure 1(a) on the powder bed, re-coater streaking Figure 1(b) and minor defects Figure 1(c).

(a)

(b)



(c)

Figure 1 Defect Images

It must be mentioned that these defects do not necessarily result in a part or build failure but could serve as an indication of a critical failure. A Failure Mode and Effects Analysis (FMEA) must still be performed to determine what types of defects are critical to the overall success of a build and which are not. However, this is beyond the scope of this research study and will not be discussed as part of this study. Object detection ML models usually perform at its best when trained with images that contain the objects or features to be detected under various scenarios or with other forms of “noise” present in the training images [19]. However, the images that were captured containing defects often only had one defect present. The rest of the powder bed around the defect only appears as a smooth grey surface. This means that while a defect can be easily isolated, it does not represent the ideal conditions to train an ML model. However, it can be hypothesised that since this is the same type of conditions under which this model will be applied, it should not be a problem. Table 3 provides the number of images for each type of defect that was used for training, as well as the number of images used for testing of the model to ensure that the training process has succeeded. As discussed previously some images may contain more than one type defect per image.

Table 2 Defect Image Dataset

Defect Type	Training Set	Testing Set
Debris	182	63
Re-coater Streaking	69	28
Minor Defect	156	61
Total Number of Images	302	108

The images used for the training and testing of the model was captured as part of a previous research study [20]. The images were captured on a Voxeljet VX500 binder jetting AM machine printing in a PMMA material. A total of 4 build jobs were monitored, capturing a photo of the powder bed after each recoating cycle. The three most prominent defects during the 4 builds are demonstrated in Figure 1. The resolution of these images are 3264 x 2448 pixels and was captured using a Raspberry Pi V2 No-IR camera module using the standard factory fitted lens. The images were used as captured and were not resized or processed in order to retain as much detail on the image as possible.

3.2 Training of the ML models

A software platform called Tensorflow was used for the training and implementation of the discussed ML model. Tensorflow is an end-to-end open source software platform that contains a variety of machine learning models and techniques [21]. The software platform was developed by the Google Brain deep learning artificial intelligence research team. The Tensorflow platform can be used for software development in Python, C++ and Javascript programming languages. The scripts used to isolate the defects in the images and for training the ML model was written using Python 3 and run inside an Anaconda virtual environment.

The computer that was used for the training of the model as well as the processing of the images had the following specifications: AMD Ryzen 5 3600 CPU, 16GB DDR4 RAM, RTX2070 Super GPU. The Tensorflow software platform also supports the use of a graphical processing unit (GPU) for the training and application of ML models. GPUs have proven their capabilities in the Machine Learning field because of their excellent capabilities in handling complex mathematical calculations. Unfortunately, several of the ML software platforms such as Tensorflow only supports CUDA enabled graphics cards [21] which is manufactured by Nvidia.

The training process was initiated for the model and took 41000 steps to reach a consistent training loss level of less than 0.05 [19]. This loss value varies between different types of models, but for this specific type of model this training loss value is considered and acceptable indicator that the model has reached a sufficient level of learning for the given training dataset.

4. MODEL EVALUATION

Once the model has been trained on the training images dataset, it is ready to be evaluated. The evaluation process is used to determine the effectiveness of the model to identify the trained objects or features from images that was not used as part of the training dataset. This process of model evaluation is important when working with ML models as the evaluation process is used to determine whether the model has reached an adequate state of learning or have possibly gone into a state of over or under fitting. The process of evaluation for an object detection model can be performed by using the mAP metric or confusion matrix. The mAP metric can be calculated using an evaluation dataset, and a confusion matrix can be drawn up using the same evaluation dataset or even an actual production dataset. As discussed previously the evaluation dataset consisted of 108 images that contained all the various types of defects previously discussed. Some images also contained all 3 types of defects on the same image. For the purposes of this study both the mAP metric will be calculated for the evaluation dataset and a confusion matrix will also be drawn up to illustrate the performance of the model graphically. The Tensorflow model evaluation script that was used to evaluate the trained model against the evaluation dataset and calculated both the mAP metric as well as the confusion matrix. The only additional process that had to be performed afterwards was to draw the confusion matrix graphically for easier analysis. **Table** displays the precision and recall results that was recorded for the Faster-RCNN model after being evaluated on the test dataset.

Table 3 Precision and Recall for trained model

Category	Precision	Recall
Debris	82.258%	80.952%
Streak	72.973%	93.103%
Minor	68.831%	85.484%

After evaluation of the model the overall mAP metric for the trained model was calculated using Equations 3 at 31.865. When comparing this to the original values as retrieved from the Tensorflow model zoo, the mAP metric calculated after the evaluation process is very close to the original value of 32. This would give an indication that the model has achieved an adequate level of learning from the training dataset.

Figure 2 demonstrates the confusion matrix that was drawn up for the model from the after processing the evaluation dataset. A confusion matrix is used to compare the predictions made by the ML model to the actual pre-labelled test images. The rows are used to indicate the different pre-labelled defects and the columns shows the predictions made by the ML model. The integer values on the matrix indicate the number of images for that category, and the percentage value indicates the percentage of all the images scanned that fell under that specific category.

Figure 2 Model Confusion Matrix

The confusion matrix indicated the type of defects with which the ML model had a higher prediction accuracy and also with which types of defects it had a lower prediction accuracy. The model struggled with the minor defects even though it consisted of the biggest image dataset. When examining both the precision and recall values for all the different classes (types) of defects, a conclusion can be made that the model had an overall accuracy of 74.687% and a recall rate of 86.513%. This is a high accuracy rate given the small training dataset but proves that the model was indeed able to detect and classify different types of defects. However, not all the types of defects fared as well. This would indicate that the model possibly might need more images for training of that specific type of defect. Figure 3 demonstrates the output image, after being processed by the ML model. The software displays both the type and level of confidence of a detected defect.

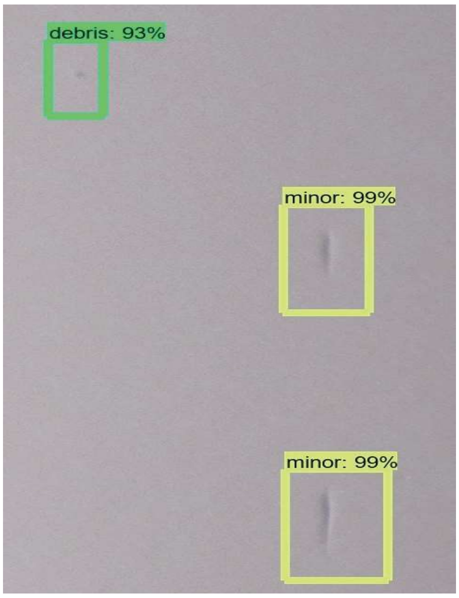


Figure 3 Detected defects

5. CONCLUSION

In this research paper the suitability of using an object detection ML model for an additive manufacturing application was investigated. The proposed idea was to use the model to detect and classify re-coating defects that occur on the powder bed surface. Literature has shown that image classification ML model could be used to detect and classify defects on a powder surface. In this research study a slightly different approach was taken to detect and classify recoating defects. A pre-trained object detection model was trained using a dataset of re-coating defects. This model was then applied to an entirely new set of images that were captured during 4 Voxeljet VX500 builds. From this evaluation process it was determined that the model had an accuracy of 75% and a recall of 87%. The model took 3 seconds per image to completely analyse the image and supply the coordinates of the defect as well as the defect type. The Faster-RCNN model has proved to be successful in identifying, localising and classifying powder bed defects as soon as they appear on the powder bed surface.

Comparing the results of this study against the results obtained in literature, the MsCNN model used by Scime and Beuth achieved an accuracy of 82% and took 7 seconds for the MsCNN to classify the image and 4 seconds for the bag of visual word classifier. For this study, an accuracy of 75% was achieved and took 3 seconds per image to process. This means that the previous authors did have a model that had superior accuracy, but it was significantly slower. It must also be mentioned that this study made use of significantly larger image files compared to the previously mentioned study as the authors only made use of grayscale image of 1280 x 1024 pixels as compared to the 3264 x 2448 pixels used for this study. The processing of these size images come with a significant computing burden, but the object detection model did provide a higher processing speed of 3 seconds per image. The processing speed of the model exceeded the predicted performance of the model as highlighted in Section 2.1. As discussed in the same section, it is very difficult to theoretically calculate the speed at which a model will process an image as there are several factors that can influence the speed. Thus, the most reliable method to determine the speed of the model is by testing it with real images. If the accuracy of the model with the analysis of minor defects can be improved, it may be possible to provide comparative accuracy results at a much higher speed.

Further build evaluation is also still needed, however due to restricted access to these machines during the 2020-year COVID-19 lockdown period precluded a detailed case study from being performed. Some of the future work that will be looked at is the optimization of the ML model to attempt to increase the accuracy as well as conducting a full case study on a build job to determine the model's effectiveness under real-world conditions and in near real time.

REFERENCES

[1] **Seifi, M., Gorelik, M., Waller, J., Hrabe, N., Shamsaei, N., Daniewicz, S. and Lewandowski, J.J.** 2017. Progress Towards Metal Additive Manufacturing Standardization to Support Qualification and Certification. *The Journal of The Minerals, Metals & Materials Society*, 69(3), pp. 439-455.

[2] **Craeghs, T., Clijsters, S., Kruth, J.P., Bechmann, F. and Ebert, M.C.** 2012. Detection of Process Failures in Layerwise Laser Melting with Optical Process Monitoring. *Physics Procedia*, 39, pp. 753-59.

[3] **Abdelrahman, M., Reutzel, E.W., Nassar, A.R. And Starr, T.L.** 2017. Flaw detection in powder bed fusion using optical imaging. *Additive Manufacturing*, 15, pp. 1-11.

[4] **Li, Z., Liu, X., Wen, S., He, P., Zhong, K., Wei, Q., Shi, Y. and Liu, S.** 2018. In situ 3d monitoring of geometric signatures in the powder-bed-fusion additive manufacturing process via vision sensing methods. *Sensors*, 18(4), pp. 1180.

[5] **Zhang, Y., Fuh, J.Y.H., Ye, D. and Hong, G.S.** 2019. In-situ monitoring of laser-based PBF via off-axis vision and image processing approaches. *Additive Manufacturing*, 25, pp. 263-74.

[6] **Southon, N., Stavroulakis, P., Goodridge, R. and Leach, R.** 2018. In-process measurement and monitoring of a polymer laser sintering powder bed with fringe projection. *Materials & Design*, 157, pp. 227-34.

[7] **Scime, L. And Beuth, J.** 2018. Anomaly detection and classification in a laser powder bed additive manufacturing process using a trained computer vision algorithm. *Additive Manufacturing*, 19, pp. 114-26.

[8] **Scime, L. and Beuth, J.** 2019. Using machine learning to identify in-situ melt pool signatures indicative of flaw formation in a laser powder bed fusion additive manufacturing process. *Additive Manufacturing*, 25, pp. 151-65.

[9] **Full Scale.io.** 2019. *Machine Learning in Computer Vision,* [ONLINE], Available at: https://fullscale.io/machine-learning-computer-vision/ [Accessed 8 June 2020]

[10] **Okafor, E., Pawara, P., Karaaba, F., Surinta, O., Codreanu, V., Schomaker, L. and Wiering, M.** 2016. Comparative study between deep Learning and Bag of Visual Words for Wild-Animal Recognition, *2016 IEEE Symposium Series on Computational Intelligence (SSCI),* pp. 1-8

[11] **Banadaki, Y., Raxaviarab, N., Fekrmandi, H. and Sharifi, S.** 2020. Toward Enabling a Reliable Quality Monitoring System for Additive Manufacturing Process using Deep Convolutional Neural Networks, [ONLINE], Available at: https://arxiv.org/abs/2003.08749 [Accessed 23 May 2020]

[12] **Craeghs, T., Clijsters, S., Kruth, J.P., Bechmann, F. and Ebert, M.C.** 2012. Detection of Process Failures in Layerwise Laser Melting with Optical Process Monitoring. *Physics Procedia,* 39, pp. 753-759

[13] **Arlen, T.C.** 2018. *Understanding the mAP Evaluation Metric for Object Detection,* [ONLINE], Available at: https://medium.com/@timothycarlen/understanding-the-map-evaluation-metric-for-object-detection-a07fe6962cf3 [Accessed 10 June 2020]

[14] **Hui, J.** 2018. *Object detection: speed and accuracy comparison (Faster R-CNN, R-FCN, SSD, FPN, RetinaNet and YOLOv3),* [ONLINE], Available at: https://medium.com/@jonathan_hui/object-detection-speed-and-accuracy-comparison-faster-r-cnn-r-fcn-ssd-and-yolo-5425656ae359 [Accessed 10 June 2020]

[15] **Tensorflow Github.** 2020. *Tensorflow detection model zoo,* [ONLINE], Available at: https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/detection_model_zoo.md [Accessed 29 May 2020]

[16] **Maheswari, J.P.** 2018. *Breaking the curse of small datasets in Machine Learning: Part 1,* [ONLINE], Available at: https://towardsdatascience.com/breaking-the-curse-of-small-datasets-in-machine-learning-part-1-36f28b0c044d [Accessed 10 June 2020]

[17] Alencar, R. 2019. *Dealing with very small datasets,* [ONLINE], Available at: https://www.kaggle.com/rafjaa/dealing-with-very-small-datasets [Accessed 10 June 2020]

[18] Du Rand, F. 2018. *DEVELOPMENT OF AN ADDITIVE MANUFACTURING RE-COATER MONITORING SYSTEM FOR POWDER BED FUSION SYSTEMS.* Magister Technologiae,Vanderbilpark: Vaal University of Technology.

[19] Juras, E. 2019. *How To Train an Object Detection Classifier for Multiple Objects Using TensorFlow (GPU) on Windows 10,* [ONLINE], Available at: https://github.com/EdjeElectronics/TensorFlow-Object-Detection-API-Tutorial-Train-Multiple-Objects-Windows-10 [Accessed 29 May 2020]

[20] Du Rand, F., Van Tonder, P.J.M., vZ Pienaar, H.C., de Beer D.J. 2018. REVIEW OF AN ACTIVE RE-COATER MONITORING SYSTEM FOR POWDER BED FUSION SYSTEMS, *RAPDASA 2018 Conference Proceedings,* pp 92-103.

[21] Abadi, M., Agarwal, A., Barham, P., Brevdo, E. et.al 2015. *Large-Scale Machine Learning on Heterogeneous Distributed Systems,* [ONLINE], Available at: https://www.tensorflow.org/about/bib [Accessed 29 May 2020]