Training Curriculum

for

**Car Parts and Damage/Defect Detection Project**

Submitted by



**Dated: 20th September 2021**

Version# 1.0

Copyright © 2021

*The Copyright of this work is vested in TPRI Technologies. This document along with all enclosures is issued in confidence for the purpose only for which they are produced. These must not be reproduced in whole or in part, or stored in a retrieval system or transmitted in any form or by any means, electronic or mechanical including photocopying and recording or otherwise for any purpose other than under agreement or with the consent in writing of TPRI Technologies.*

**CHANGE HISTORY**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sl. No | Version | Date | Changed By | Description |
| 1 | V1.0 | 20-09-2021 | Ishan Borkar | Original |
|  |  |  |  |  |

All enquiries with respect to this document should be referred to:

|  |  |  |
| --- | --- | --- |
| **Name** | **Designation** | **Contact Details** |
| Pavan Kulkarni | Project Lead | Email: pavan.kulkarni@technoproindia.com  Mobile: +91-9902762074 |

**Table of Contents**

[1. Industry 4.0 5](#_Toc80085410)

[1.1. AI in Industry 4.0 5](#_Toc80085411)

[2. Car Parts and Defect Detection/Localization 7](#_Toc80085412)

[3. Car Parts Detection 8](#_Toc80085413)

[3.1. Yolov3 Architecture 8](#_Toc80085414)

[3.2. Input of Bounding Box for Car Parts Annotation 10](#_Toc80085415)

[3.3. Results of Car Parts Detection 10](#_Toc80085416)

[3.4. Results using Average Confidence Score for Car Parts Detection 11](#_Toc80085417)

[3.5. Performance Measurement 12](#_Toc80085418)

[3.6. Market Study of Car Parts Detection 13](#_Toc80085419)

[4. Car Damage Detection 15](#_Toc80085420)

[4.1. MaskRCNN Architecture 15](#_Toc80085421)

[4.2. Detectron2 Architecture 16](#_Toc80085422)

[4.3. Input of Bounding Box for Car Damaged Parts Annotation 16](#_Toc80085423)

[4.4. Results of Car Parts Damage Detection 18](#_Toc80085424)

[4.5. Results of Average Predicted IOU Score for Car Parts Damage Detection 19](#_Toc80085425)

[4.6. Market Study of Car Parts Damage/Defect Detection 19](#_Toc80085426)

[5. Conclusion 20](#_Toc80085427)

[6. References 21](#_Toc80085428)

# **Industry 4.0**

Introduction: Industry 4.0 is the digital transformation of the manufacturing / automotive industry or related industries. It is the intelligent networking of machines and processes with the help of information and communication technology. For this to happen we require modern control systems, embedded software along with IoT (Internet of Things) which are integrated with machines and processes which enables a new way of production, value creation and real time optimization.

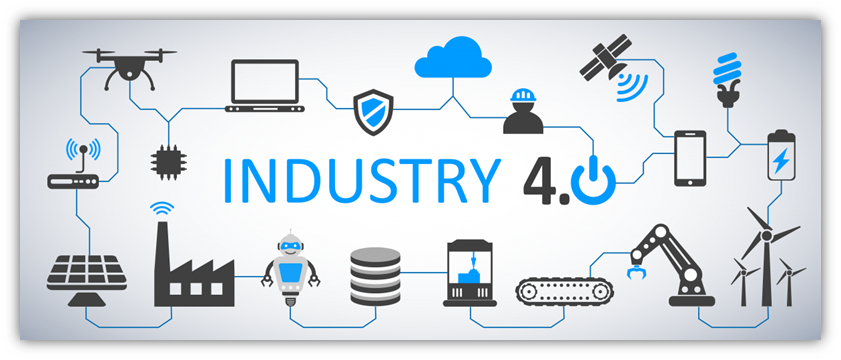


Figure 1: Industry 4.0

## **AI in Industry 4.0**

One of Artificial Intelligence's benefits is that it allows robots to learn from the data they collect from industry activities, allowing them to improve their skills in every engagement. After all, this is one of Industry 4.0's essential pillars, and it will help industries become more autonomous and productive.

Three advantages of AI for the manufacturing

When used correctly, artificial intelligence has a number of benefits for the manufacturing business. we've compiled a list of three of the most important benefits for the manufacturing industry today:

1. Error reduction: After being trained, intelligent algorithms can perform very well with jobs that are prone to human errors.
2. Cost reduction: Robots are being used by a number of e-commerce sites and banks to commence client assistance. If the problem is more complicated, the human attendant is summoned. Companies can cut personnel costs or reassign people to more strategic responsibilities, allowing them to enhance profits and focus on their core business.
3. Revenue Growth: Decision-makers will have more time to think about the core processes and leave other AI tasks to personnel who make fewer mistakes and focus on more vital operations.

By implementing the AI, the production engineers will be able to identify what deficiencies are there in the production lines which in turn gives the better-quality product and hence better yield.

As per the statistics [1],

1. 71% of CEOs believe AI will have a substantial influence on their company.
2. 50% of enterprises within the next 5 to 7 years may double their cash that rely on AI.
3. 64% industrial firms have already begun to invest in AI solutions.
4. $ 3.7 trillion Artificial Intelligence will contribute to the manufacturing industry by 2035.
5. 36% of Industrial companies’ integration and compatibility problems are faced with AI solutions.

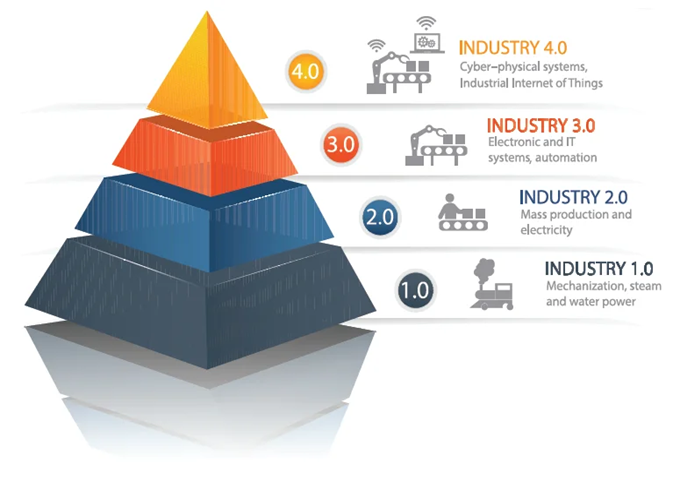


Figure 2: Evolution of Industry

# **Car Parts and Defect Detection/Localization**

In this problem statement, we try to detect the car part and its damage in an assembly line automation. It is categorized as Segmentation. There are different kinds of deep learning techniques which can be used to solve this problem. They are CNN (Convolutional Neural Network), RNN (Recurrent Neural Network), FRCNN (Fast Recurrent Neural Networks), MaskRCNN (Mask Recurrent Neural Networks).

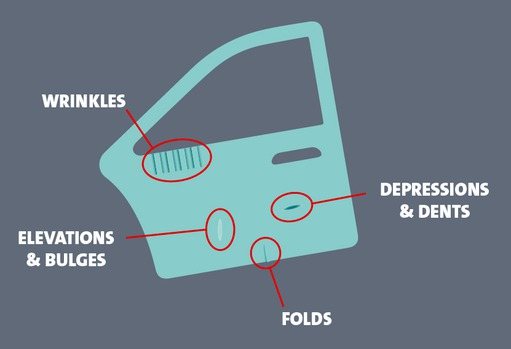
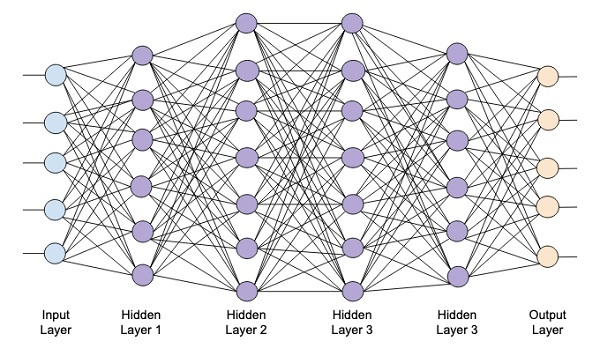
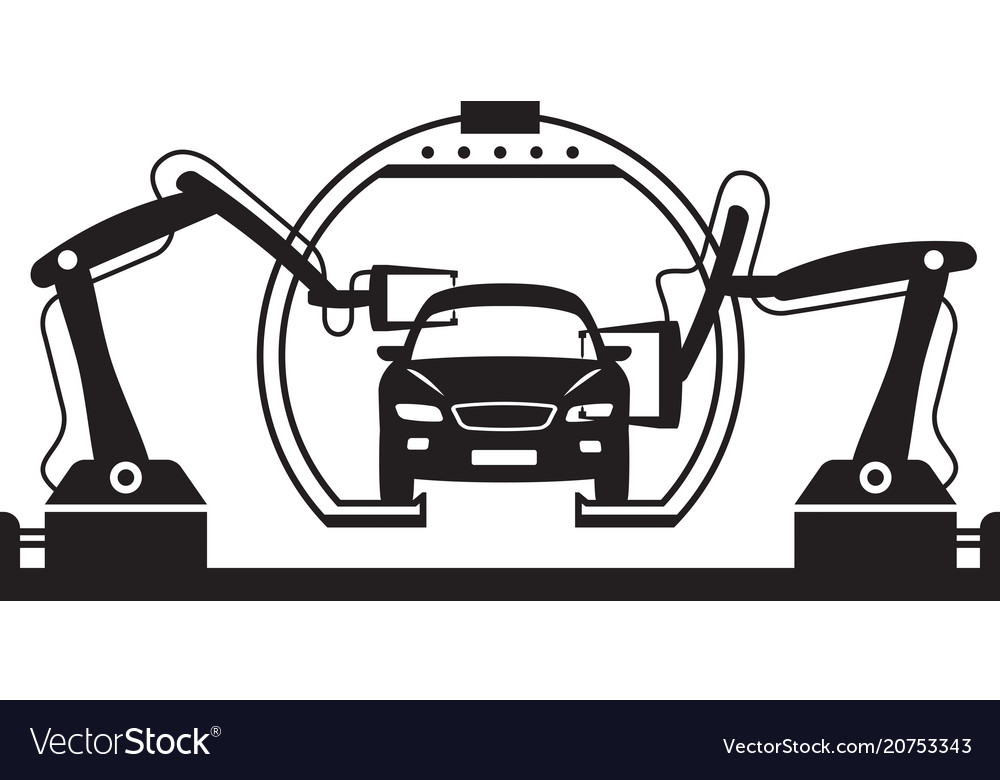


Figure 3: Detecting Damages on a car part assembly using neural network

In this figure, an image of the car parts from the assembly line production is taken. It is then sent to the neural networks which does the work of classification of parts based on its damages and detects it using a bounding box. The neural network acts like a human brain. It has three layers. 1. Input layer, 2. Multiple Hidden layers and 3. Output layer. The output is then obtained in the form of labels which is used to recognize the type of damages like wrinkles, depressions and dents, folds, elevations and bulges, etc.

# **Car Parts Detection**

## **Yolov3 Architecture**

For detecting the car parts, we use the Yolov3 algorithm. The architecture for this algorithm is as follows:

You only look once (YOLO) is a state-of-the-art, real-time object detection system [2].

The network has 24 convolutional layers followed by 2 fully connected layers. 1×1 reduction layers are used followed by 3×3 convolutional layers. Before detection, it repurposes classifiers and localizers to perform detection. It applies the model to an image at multiple locations and scales. A single neural network is applied to the whole image. This network divides the image into regions and predicts the bounding boxes and probabilities for each region. The bounding box prediction is done by predicted probabilities. The final output of the network is a 7 × 7 × 30 tensor of predictions.

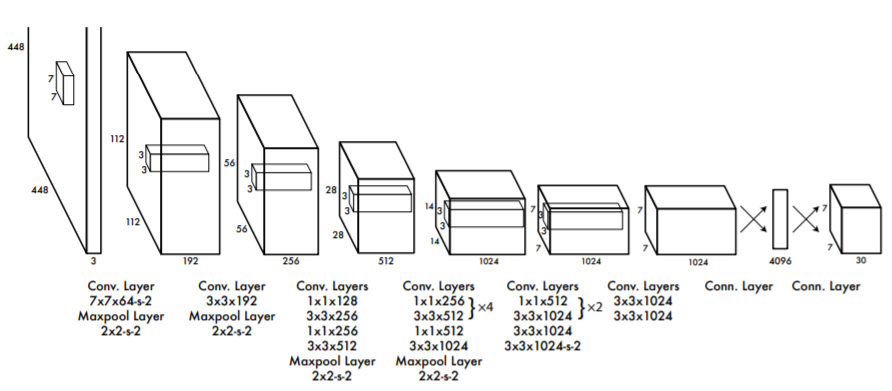


Figure 4: Yolov3 Detection Architecture [4]

**Method**:

1. Training the yolov3 model: Run the following codes:

1. Minimal\_Example.py: It includes the car pictures with the bounding boxes with confidence scores and a csv file with the name and location of the bounding boxes.
2. Convert\_to\_YOLO\_\_format.py: It prepares the dataset in the yolo format to train the model.
3. Download\_and\_Convert\_YOLO\_weights.py: It downloads the pre-trained yolov3 weights and converts them to the keras format.
4. Train\_YOLO.py: It runs the YOLOv3 model with the help of keras and tensorflow packages and creates a YOLOv3 model with 9 anchors (anchor boxes) and 5 classes.

2. Run the Inference: In this step, we perform object detection on the new images. It displays the name of the part on the bounding box and IOU (Intersection over union) score of the area covered by the bounding box for that particular part. If IOU score is greater than 0.5, it is a good prediction. To apply Intersection over Union to evaluate an object detector we need:

1. The ground-truth bounding boxes
2. The predicted bounding boxes from our model

Formula for IOU score is IOU=Area of Overlap/Area of Union



Figure 5: The predicted bounding box is in red colour and the ground-truth bounding box is in green colour [5]

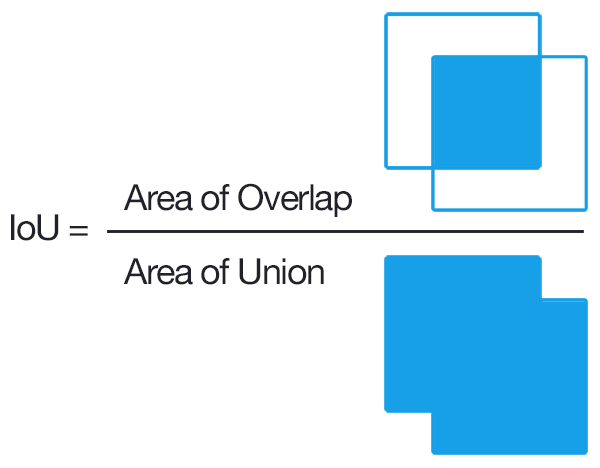


Figure 6: Formula for IOU

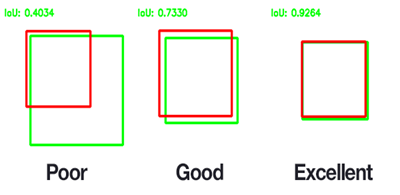


Figure 7: Different kinds of IOU

Advantages of YOLOv3 Algorithm:

1. It takes the whole image during the test time so that its predictions are informed by global context in the image.
2. It makes the prediction with a single network evaluation.
3. It is fast, more than 1000 times faster than R-CNN and 100 times faster than Fast R-CNN.
4. It is much faster and more accurate than YOLO.

## **Input of Bounding Box for Car Parts Annotation**

In this step, the car data taken from [7] needs to be trained. In order to perform training, we first label it using an open-source annotation and labelling tool called VoTT (Visual Object Tagging Tool). Different parts of the car like Glass, Side Glass, Wheel, Light, Door are being annotated by the annotators.

 7

Figure 10: Annotated parts of car

## **Results of Car Parts Detection**

After the annotation step is complete, we train our Yolov3 algorithm to detect all the car parts which are visible with the predicted IOU score on the bounding box. We have obtained results for training data as follows:

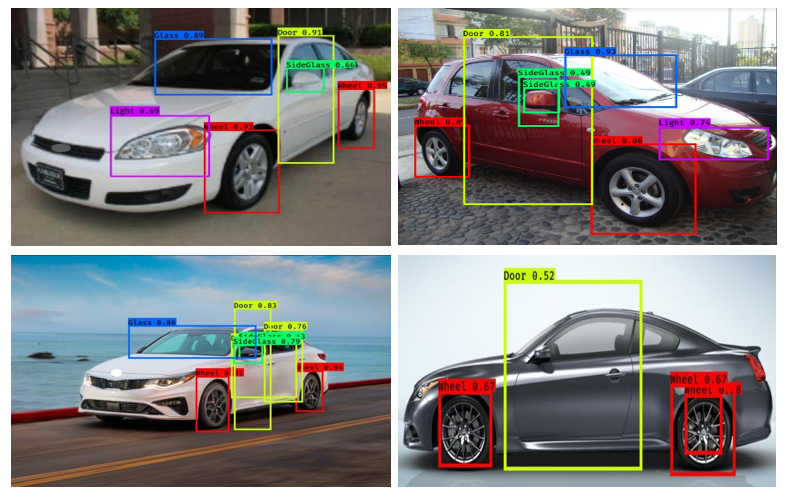


Figure 11: Object Detection on the training data

The same algorithm is used for the test data which is unseen data in which we do not annotate or label these images. They are directly being used by the algorithm to perform object detection. The results are as follows:

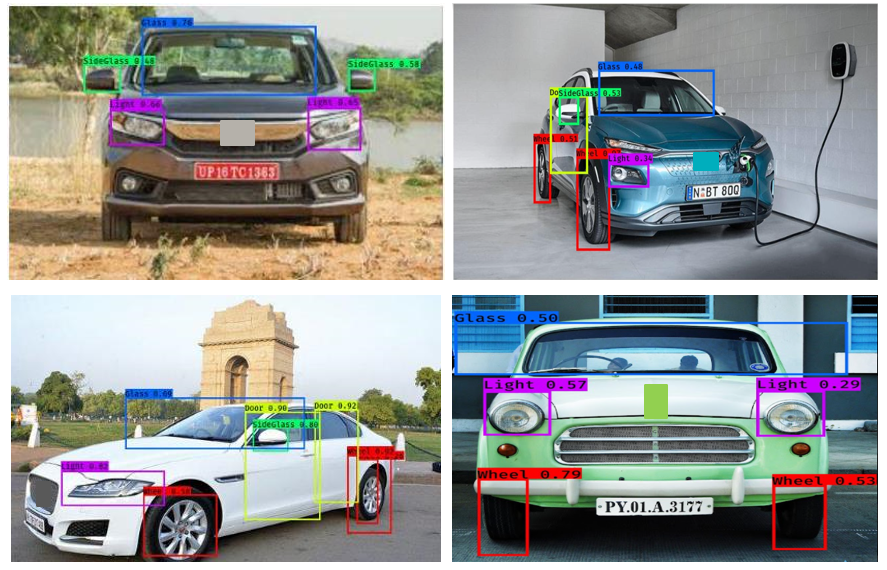


Figure 12: Object Detection on the test data

## **Results using Average Confidence Score for Car Parts Detection**

The below table represents the average confidence score/IOU of each label in every phase of this project. We calculate the average predicted score for each phase.

The project was performed in three phases. They were

1. **Phase 1**: In this phase, the model was trained using the car image dataset from [7] and tested using the car image dataset from [7]. The image annotation was performed on the training images using VoTT Annotation tool.

2. **Phase 2**: In this phase, the model was trained and tested using the 2 newly captured car images from Google and included in the directories mentioned in Phase 1. The image annotation was performed on these 2 images using VoTT Annotation tool.

3. **Phase 3**: In this phase, the model was only tested using 10 newly captured images from Google. The images were neither annotated not they had undergone the training model.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Label Index** | **Label** | **Confidence Score in Phase 1** | **Confidence Score in Phase 2** | **Confidence Score in Phase 3** |
| 0 | Light | 0.591444 | 0.618585 | 0.511663 |
| 1 | Glass | 0.745640 | 0.872962 | 0.716830 |
| 2 | Side Glass | 0.490936 | 0.506028 | 0.490544 |
| 3 | Door | 0.638261 | 0.667751 | 0.586294 |
| 4 | Wheel | 0.746288 | 0.925647 | 0.794101 |
|  | **Average Confidence Score** | 0.660311 | 0.669127 | 0.640702 |

Table 1: Average Confidence score for Car Parts Detection from phase 1 to phase 3

The average execution time is less than 2 seconds per image.

## **Performance Measurement**

We are calculating the performance with respect to the following parameters:

1. **Total Loss**

We see from the graph of Total Loss vs No. of Epochs, that our model is achieving the desired results after performing 500 epochs.

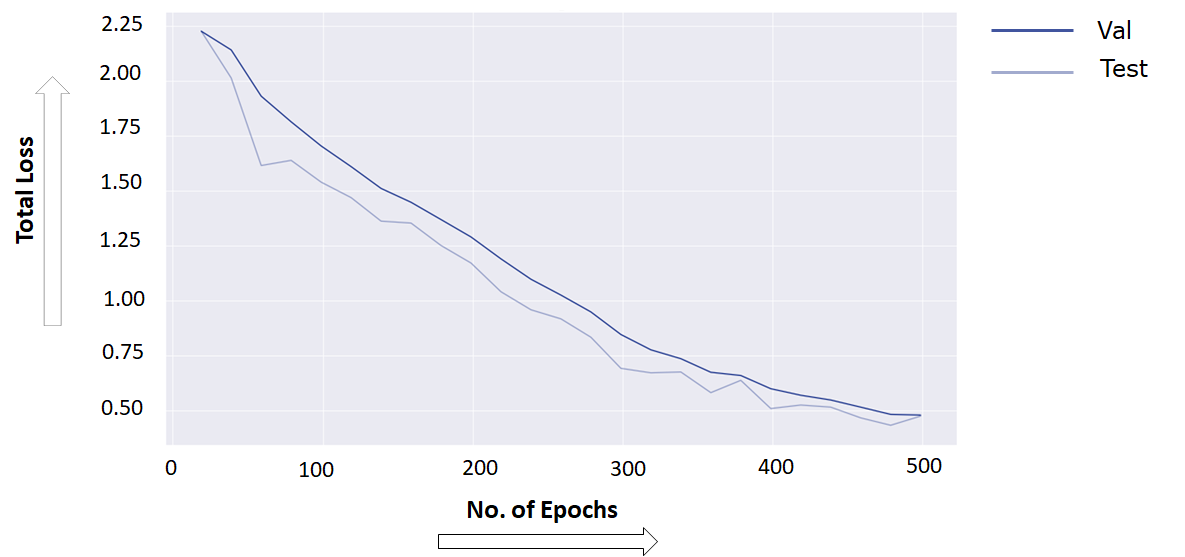
****

Figure 13: Total Loss vs Number of Epochs

1. **Class Accuracy**

Similarly, we see from the graph that the accuracy is improving with respect to the number of epochs as we increase it.

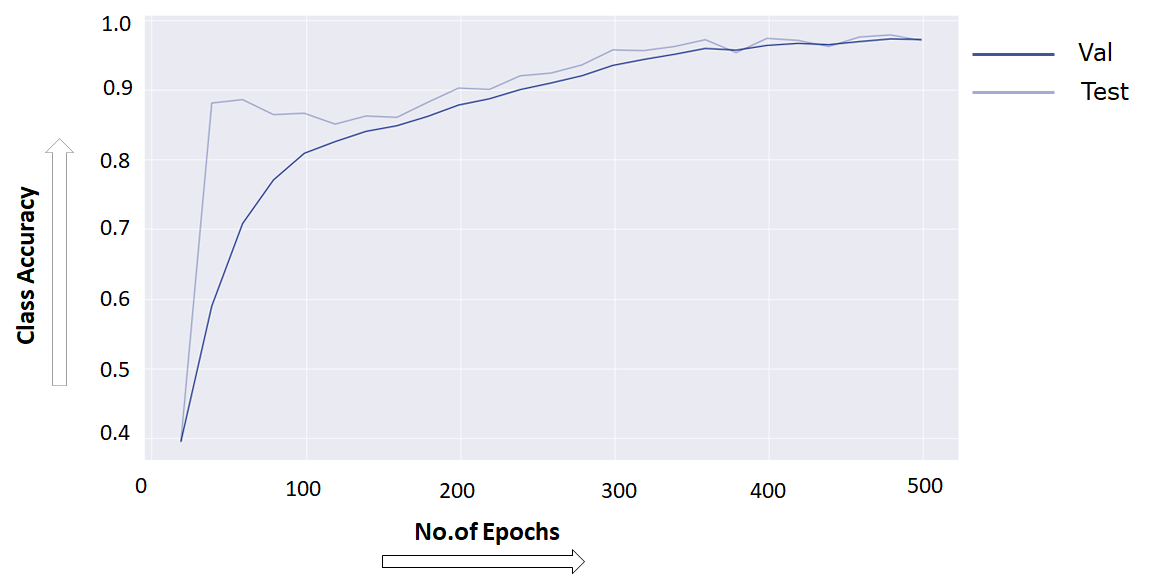


Figure 14: Class Accuracy vs Number of Epochs

## **Market Study of Car Parts Detection**

Yolov3 is a real-time object identification technique which uses the neural networks to perform the task. It is very much popular due to its speed and precision. This can be used in a wide range of applications like identifying traffic signals, parking meters, animals, people following COVID-19 Standard Operating Procedures (SOPs) like social distancing, wearing masks, etc. In our case, we have used it for identifying different parts of the car based on the images obtained. We are targeting some industries like Autonomous Vehicles, Automotive, Healthcare, Steel Plants to use this algorithm for Object Detection.

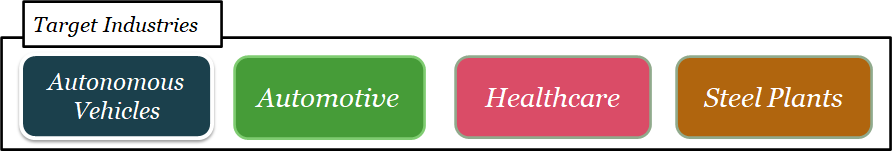


Figure 15: Target Industries for Object Detection

# **Car Damage Detection**

For detecting the car damage, we use MaskRCNN and Detectron algorithm first we will explain about the MaskRCNN architecture and later on Detectron2 architecture will be explained.

## **MaskRCNN Architecture**

We begin with an image that is sent via convolutional layers or a backbone to obtain feature maps on one side and anchor boxes on the other, all of which are sent to the region proposal network (RPN). The region proposal network goes into Region of Interest (ROI) pooling with the feature maps. It means that the regions which were proposed by this RPN original proposal network, and the output of this with the feature maps will go to the ROI Pooling layer. Finally, we will get the classes using ROI pooling, so there is a classification sublayer that tells us the type of the object or if it is a background, it is a box regressor, and the other branch is used for classifying pixels, which gives us the segmented parts, so all three branches are active at the same time [6].

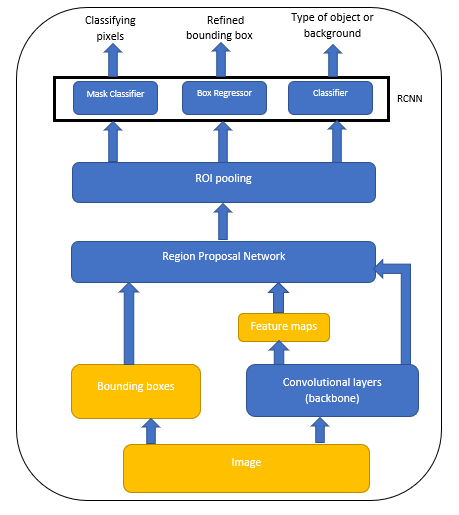


Figure 8: Architecture of Mask RCNN

## **Detectron2 Architecture**

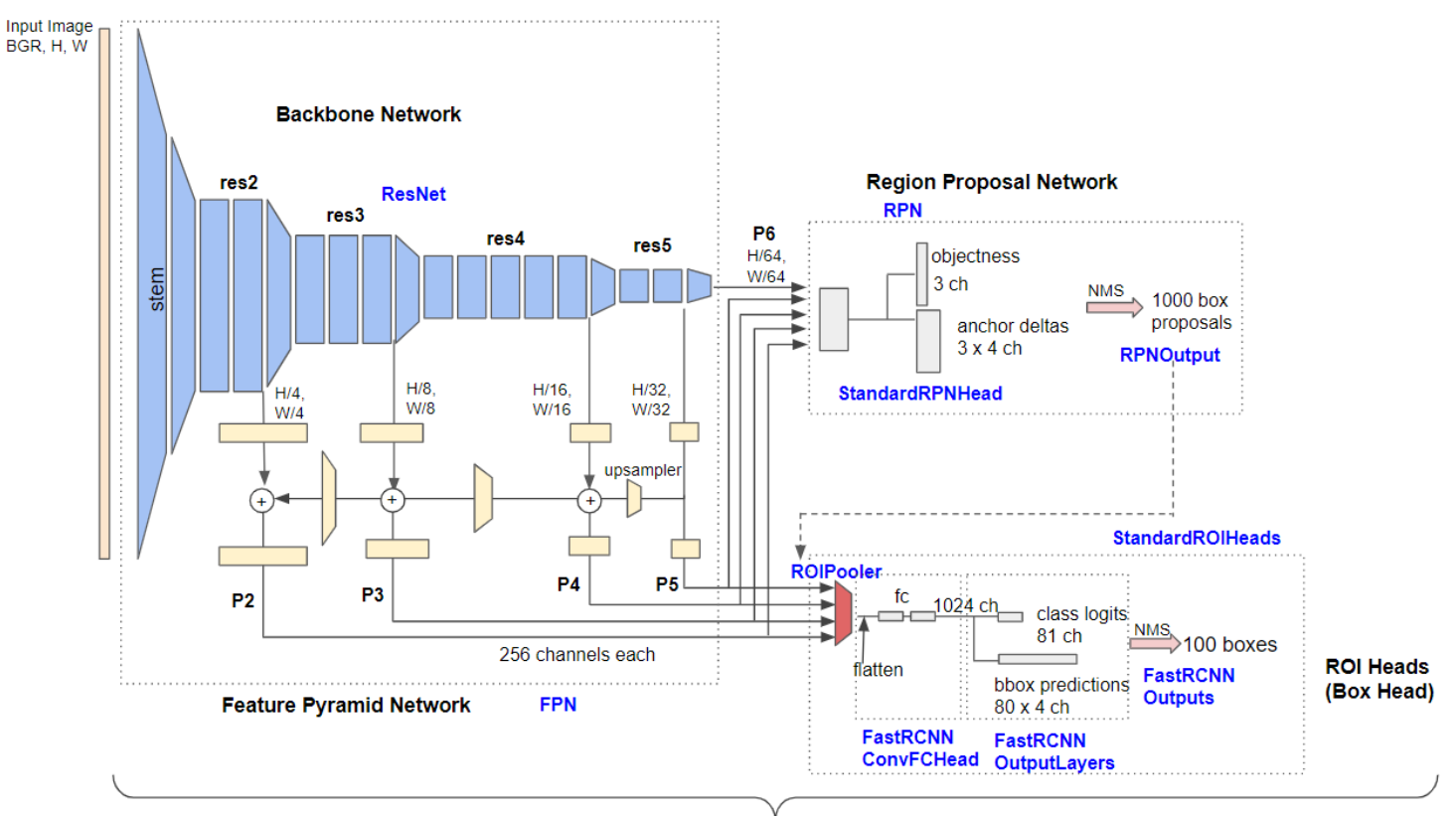


Figure 9: Architecture of Detectron 2

Facebook AI Research's Detectron 2 is a next-generation open-source object detection system. You may use and train state-of-the-art models for detection tasks including bounding-box identification, instance and semantic segmentation, and person key point detection using the repository. The different parts include:

1. Backbone Network: It extracts feature maps at various scales from the input image. P2 (1/4 scale), P3 (1/8), P4 (1/16), P5 (1/32) and P6 (1/64) are the output features of Base-RCNN-FPN (Base - Recurrent Convolutional Neural Networks- Feature Pyramid Network). The output feature of non-FPN (‘C4') architecture is only from the 1/16 scale.
2. Region Proposal Network (RPN): It uses multi-scale characteristics to detect object regions. By default, 1000 box proposals with confidence scores are generated.
3. ROI Box Heads: It cuts and warps feature maps into several fixed-size features using proposed boxes, and achieves fine-tuned box positions and classification results using fully-connected layers. Finally, using non-maximum suppression (NMS), 100 boxes (by default) are filtered out. One of the sub-classes of ROI Heads is the box head. Mask R-CNN, for example, features additional ROI heads, such as a mask head.

## **Input of Bounding Box for Car Damaged Parts Annotation**

In this step, the data of damaged cars from [8] needs to be trained. In order to perform training, we first label it using an open-source annotation and labelling tool called VoTT (Visual Object Tagging Tool). Different damaged parts of the car like Broken Glass, Bumper Dent, Door Dent, Scratch, Smash are being annotated by the annotators.

****

Figure 16: Annotated damaged parts of car

**Input for Car Parts Damage Detection using algorithms:**

1. **Detectron2:** In this problem statement we capture the open-source dataset. We initialize the instance annotations using the json file of train annotation and multiple train annotation. We receive the different categories and super-categories images in the form of damage parts. E.g., headlamp, rear bumper, door, hood, front bumper. They are then mapped using a dictionary for category id and category name.
2. **MaskRCNN:** We specify image id, image, mask, class id, bounding box. There are 2 classes: BG (Background), 2) Damage. We also run a json file via region data to get the details of each image we want to train for detecting damages.

**Output for Car Parts Damage Detection using algorithms:**

1. **Detectron2:** We get the plots for the annotations on different damaged parts in the form of bounding boxes and segmentation. When we run the Detectron2 libraries for training the model, we receive results like total number of damage instances, Average Precision, Average Recall, IOU score for each prediction, class accuracy, total loss. We also run the validation images to get the desired results of the damages in the form of IOU scores above the bounding box.
2. **MaskRCNN:** We receive the results of instance segmentation for the damaged parts separating foreground and background along with the bounding box for annotations. Visualization in the form of histogram for the model weight matrix is carried out in the form of descriptive statistics. We also run the training and validation images to get the desired results of the scratches in the form of IOU scores above the bounding box.

## **Results of Car Parts Damage Detection**

First as we take the image of a damaged car from [8] which needs to be annotated, we first label it using an open-source annotation and labelling tool called VoTT (Visual Object Tagging Tool). As we annotate the car damaged parts as scratches, bumper dent and door dent etc. then these images are sent to the algorithm to learn from the annotated and labelled images.

****

Figure 17: Car Parts Damage Detection on Training data

Once the training of the algorithm is completed, then the unseen images and without annotated images are given to the algorithm, then results are obtained with masks on the different classes of damages on top of the car image.

****

Figure 18: Car Parts Damage Detection on Test data

## **Results of Average Predicted IOU Score for Car Parts Damage Detection**

The below table represents the comparative results of segmentation, bounding box, average predicted IOU score of damage/scratch for Detectron2 and MaskRCNN algorithms.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sr. No.** | **Models Used for Car Parts Damage Detection** | **Kinds of Detection** | **Separating damaged part from the image** | **Segmentation Result** | **Bounding Box Result** | **Average Predicted IOU Score** |
| 1. | Detectron2 | Damage | Not Supported | 95.7% | 98.9% | 94.46% |
| 2. | MaskRCNN | Damage, Scratch | 95.5% | 92.3% | 96.6% | 94.48% |

Table 2: Comparative Results of Segmentation, Bounding Box and Average Predicted IOU Score

The average execution time per image is 10 seconds.

## **Market Study of Car Parts Damage/Defect Detection**

MaskRCNN is a deep neural network designed to handle the problem of instance segmentation in machine learning and computer vision. This algorithm can help to distinguish between various things in an image or in a video. It segments objects and shapes from the images. It results in the object’s bounding boxes, classes and masks. This can be used in varied applications like automobile insurance companies, medical, traffic road scenes, etc. In our case we have used it to detect the different types of damages which are occurring most frequently in a car. We are targeting some industries like Autonomous Vehicles, Automotive, Healthcare, Steel Plants to use this algorithm for Object Damage or Defect Detection.

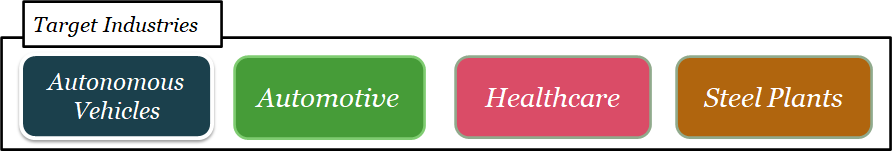


Figure 19: Target Industries for Object Damage or Defect Detection

# **Conclusion**

We are focusing on two parts: vision-based car parts detection and car damage detection and segmentation. In vision-based car parts detection, the areas which we focus on include the intelligent transportation industry like automotive assembly lines and traffic surveillance systems. In car damage detection and segmentation, the areas which we focus on include the car manufacturers and other large automotive companies, vehicle inspection, car rental agencies, insurers.

# **References**

1. <https://alfapeople.com/artificial-intelligence-ai-industry-4-0-how-microsoft-technology-is-bringing-major-changes-to-manufacturing/>
2. <https://pjreddie.com/darknet/yolo/>
3. Joseph Redmon, Ali Farhadi. “YOLOv3: An Incremental Improvement”. <https://pjreddie.com/media/files/papers/YOLOv3.pdf>
4. Joseph Redmon, Santosh Divvala, et al. “You Only Look Once: Unified, Real-Time Object Detection”. 9 May 2016, <https://arxiv.org/pdf/1506.02640.pdf>
5. <https://www.pyimagesearch.com/2016/11/07/intersection-over-union-iou-for-object-detection/>
6. Kaiming He, Georgia Gkioxari, et al. "Mask RCNN". In Proceedings of the IEEE international conference on computer vision, pp. 2961-2969. 2017, <https://arxiv.org/pdf/1703.06870.pdf>
7. Dataset for Car Parts Detection: <https://github.com/bhadreshpsavani/CarPartsDetectionChallenge/tree/master/Data/Source_Images>
8. Dataset for Car Parts Damage Detection:

<https://www.kaggle.com/lplenka/coco-car-damage-detection-dataset>

<https://github.com/nasirtrekker/DeepLearning_MaskRCNN/tree/master/car-damage-detection-using-CNN/custom>