DEEP LEARNING FOR THE DETECTION OF CAR FLAP STATES

Benoît Guérand
Karlsruher Institut für Technologie
Daimler Protics GmbH
benoit.guerand@mercedes-benz.com

Fabian Scheer
Daimler Protics GmbH
fabian.scheer@mercedes-benz.com

Mustafa Demetgül
Karlsruher Institut für Technologie
mustafa.demetguel@kit.edu

Jürgen Fleischer
Karlsruher Institut für Technologie
Juergen.Fleischer@kit.edu

ABSTRACT

In recent years, deep learning and object detection has continuously attracted more attention. Especially in the automotive world where many car manufacturers are currently investigating its possible applications. On production lines, even if processes are more and more automatized mistakes can happen and hinder the performance of an industrial plant. In this study, a method and application of object detection-based deep learning algorithm to detect open flaps on cars, like doors, trunk, hood etc. is examined. With this approach, the advantages of gap detection in cars on production lines, specifically the application of Resnet50 Convolutional Neural Networks (CNNs) and transfer learning in an industrial use case, are demonstrated. We show how the problem of detecting open flaps on cars is modeled in a way that a CNN can be applied to this new kind of application and present a detailed evaluation of the results and challenges. Finally, many suggestions are given for future applications of similar algorithms.

Keywords

deep learning, Resnet50, RetinaNet, door gaps, object detection of open car flaps, Convolutional Neural Network (CNN), industrial use case, production line

1. INTRODUCTION

Generally, assembly is done with the help of robots in automotive production lines. Misalignment of assembled parts or open parts of the vehicle body (e.g. doors, trunks, etc.) during production can cause collisions between the robots and car components. To avoid such problems computer vision techniques can be used to improve process productivity and production flexibility, provide position information to the robot controller and set or correct the robot's path [1, 2]. For this, a camera must be placed at different positions on the production line. Typical challenges of machine vision techniques applied to automotive production lines remain, like the influences of lighting, light reflections and the changing of image backgrounds. Object detection may be an important option for solving these problems [3].

Object detection is gaining a lot of attention recently as its applications cover a very large field of studies [4]. Object detection has many different application domains such as pedestrian detection, behavioral analysis, autonomous driving, face recognition, pattern recognition, car vision, and so on [5]. When the literature is examined, there are studies on the detection of car body problems like scratches with deep learning, but there is no study on the detection of open flaps or gaps on cars on the production line. Open car flaps are the doors, the trunk, the hood or the tank cap.

The purpose of this article is to contribute to the literature on the detection of open flaps or gaps on cars. This problem is of particular interest to the automotive industry as it causes potential damage to vehicles during production, and this problem has not yet been solved using object detection algorithms. A method based on Resnet50 is presented and it is

shown how to model the problem of detecting open flaps on cars as an object detection problem. An industrial use case in a real production line is used for a detailed evaluation and discussion of the presented method.

2. RELATED WORK

Looking at the literature studies on this subject, Kang [3] tried to reduce the wrong decision-making processes caused by ambient lighting and light reflection problems during the detection of problems by monitoring the automobile production line with a camera. In their studies, the distance between the car body and the door part and the door was obtained with the measuring device combining the laser slit light source and the LED patterned light source [3].

Kosmopoulos and Varvarigou in [6] introduced a system for automatic gap inspection using computer vision. It can measure the lateral and gap size of the gap. The measurement setup consists of two calibrated stereo cameras and two infrared LED lamps, which are used to highlight the edges of the range through specular reflection [6]. Considering these studies, it is applied for a single door and many tools are needed.

Mazzetto in [7] have implemented deep learning-based object detection, semantic segmentation, and anomaly detection to assist in finding automotive assembly errors. They worked on the brake, disc, and motor assembly [7].
A study has also been carried out on the determination of the outer body and interior parts of the cars with object detection. For this, Resnet 50 and Darknet are used. Although this study is close to our study, only parts of it were detected. There is no abnormal detection [8]. This is the closest study to the study we have done. However, object detection has been used in the problem detection of different vehicle parts. Rahimi in [9] used the YOLO deep learning algorithm, which distinguishes between vehicles and people at an automotive manufacturing plant using object detection [9].
Apart from these studies, detection of vehicle damages without using object detection is done with deep learning methods [10, 11]. In addition, there are studies carried out for the diagnosis of problems in different production lines [12-13]. However, this technique is not suitable for our application.
When doing quality control with images, the main issue is the effect of ambient and light reflection, the background and the doors will be in approximately the same place as the cars will be located in the same place each time. Object detection avoids using many complex filters to remove the effect of such parameters [14]. This domain doesn´t have a lot of literature so this paper will help increase resources regarding this topic. The goal of our paper was to add more literature on this specific topic as deep learning is an always growing field and such implementation was relevant for our case in the production plant.

This paper demonstrates a new approach to detect the state of vehicle flaps in production lines using the object detection-based Resnet 50 deep learning method.

3. THEORETICAL BACKGROUND

There are many algorithm architectures that are able to solve this problem [15-16]. But according to the literature [17-18] and what is the most efficient at this time; the ResNet 50 method is chosen to solve the object detection problem. ResNet50 is a special type of Convolutional Neural Network (CNN) [17-18]. CNNs are typically used the most to compute image data, as the architecture is well suited to detect patterns such as curves, lines, etc. A CNN typically consists of three layers. The first is a convolutional layer which is like a filter. It has a matrix as input and a kernel (filter). This is shown in Figure 1.

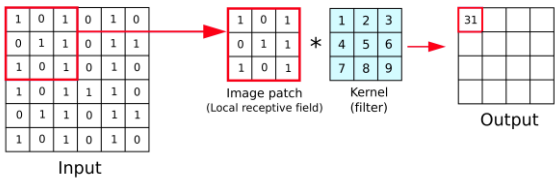

Figure 1: Convolutional layer, *similar to [19]*

Then it has the pooling layer which reduces the parameters of the input matrix such as a max-pooling layer where the maximum of each quadrant is taken (see Figure 2).

$$\begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \\ 13 & 14 & 15 & 16 \end{bmatrix} \rightarrow \begin{bmatrix} 6 & 8 \\ 14 & 16 \end{bmatrix}$$

Figure 2: Pooling

The last layer is a fully connected layer which is a feed-forward neural network.

The algorithm used in this paper is an ssd_resnet50_v1_fpn_coco (also called Retinanet

[20]). This is an assembly of various architecture and this combination was firstly introduced to have one stage proposal-driven mechanism while maintaining performances. The main component of this architecture is a ResNet50. It is a type of CNN where a convolutional layer, a pooling layer, 50 convolutional layers, an average pooling layer and a fully connected layer at the end is used. It is our backbone network.

The specificity of a ResNet is that the group of 50 convolutional layers is using "identity shortcut connections"(Figure 3). This was introduced at a time when the main method for building a network was to add more layers. But it was shown [20] that is wasn't the most effective method, as the accuracy saturated when the network was converging: this is called degradation. The ResNet architecture Figure 4 solved this problem. The novel solution introduced residual networks (shown in Figure 3). They allow to skip layers. This will permit the network to avoid the layers that are nuisances for the results during the regularization phase. This results in a very deep network without the burden of the large amount of layers.

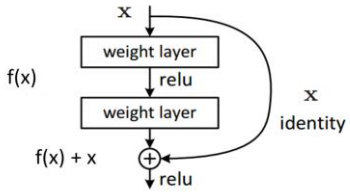

Figure 3: Resnet Structure, *similar to [21]*

The main advantage of this architecture according to its authors [21] is that you have better results than the actual network with a lower amount of parameters. It is using fewer parameters than its former counterparts, such as VGG algorithms [17]. The VGG-16 uses 134,7M parameters whereas the ResNet50 only uses 23,9M.

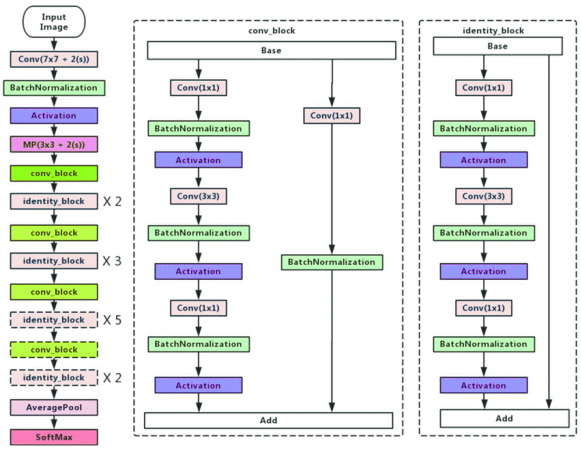

Figure 4: Resnet50 architecture, *similar to [22]*

To measure accuracy in object detection, the metric IoU is introduced, which means Intersection over Union. The formula of the IoU is the following:

$$IoU = \frac{|Area(Ground\ truth\ box) \cap Area(Anchor\ box)|}{|Area(Ground\ truth\ box) \cup Area(Anchor\ box)|}$$

An IoU bigger than 0.5 is considered as a "good" metric [23].

We have manually defined our bounding box of the original object: it is the ground-truth bounding box. Then the algorithm will generate multiple random bounding boxes with different scales and different forms: these are called anchor boxes.

In Figure 5 we have the ground truth box marked in green which shows the truth and an anchor box that is marked in orange. The intersection of both boxes is marked in light orange (Figure 5).]

Figure 5: IoU illustration

For object detection, the algorithm is fed with data consisting of pictures where the position of the object is manually annotated. These are the ground truth boxes. The coordinates of the object form a box. To

make predictions, the algorithm will guess where the boxes are. But to help find the boxes, we add some layers to the algorithm: convolutional layers. These layers are forming a so called Single-Shot Detector (SSD). Although SSD have a lower accuracy by 10% in average [20] of a two-stage method, they are designed for speed and efficiency. SSD will divide the image using a grid and will try to detect the objects in each grid. Then the SSD calculates the probability that the object is present by comparing it with predefined anchor boxes (Figure 6).

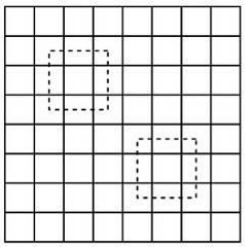

Figure 6: Object detection, *source [24]*

We adopted the feature pyramid network from [26] (Figure 7). This helps finding objects from different scales on the same image. This is very useful for our situation, where we have very small gaps for the doors and bigger gaps when the trunk is open. The main principle is to take an image and subsample it using convolutional layers to transform it into lower resolution and lower image size, hence forming a pyramidal structure while keeping the strongest features using lateral connections [20] (Figure 7).

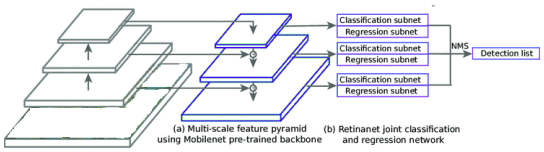

Figure 7: Feature Pyramid Network, *similar to [20]*

To achieve better results with a small amount of data, it is advisable to use a technique called transfer learning [25]. Thereby, a model is used that has already been trained on a different dataset. This allows to reach far better results with the own dataset. In our case the algorithm was trained on a dataset called COCO17 (Common Objects in Context) [29] and we trained the algorithm from this starting point.

4. EXPERIMENTAL SETUP

Our use case of detecting open flaps on cars normally is a problem in the domain of measurement technology. We reformulate this problem as an object detection problem and define separate classes for the state of each flap. To evaluate the presented method, a setup in a real automotive production line is used. We drew our process in Figure 10 where we use a classical deep learning algorithm training, optimizing process.

Three cameras mounted on a steal structure are used for the system. The cameras cover three different viewing directions onto a car (see figure 8): the front view for the hood; the left view for doors on the left side and the trunk; the right view for doors on the right side, the tank cap and the trunk again.

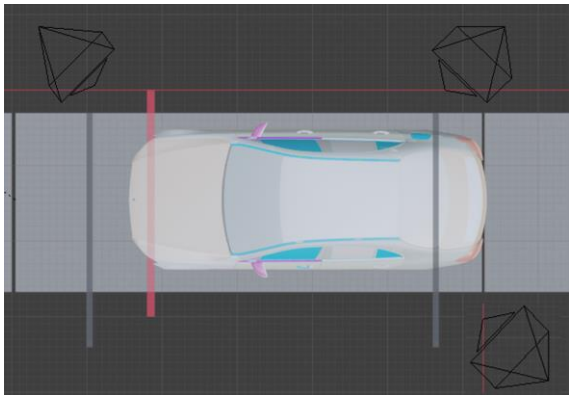

Figure 8: Camera layout

For the system, color cameras of the type UEye UI-3000SE-C-HQ with global shutter, 12 mm focal lense and pixel dimensions of 4110x3006 were used. The cars run on a conveyor belt and are thereby passing the viewing range of the cameras (Figure 9). A light trigger signals that a new car enters the station. Approximately 5 meters after the light barrier the car is in full view of the cameras and pictures for each camera are taken. This is triggered by a fix increment value of the conveyor belt that is measured by a rotary encoder. The signal of the light barrier resets this increment value to zero for each new car.

Figure 9: Different camera angles

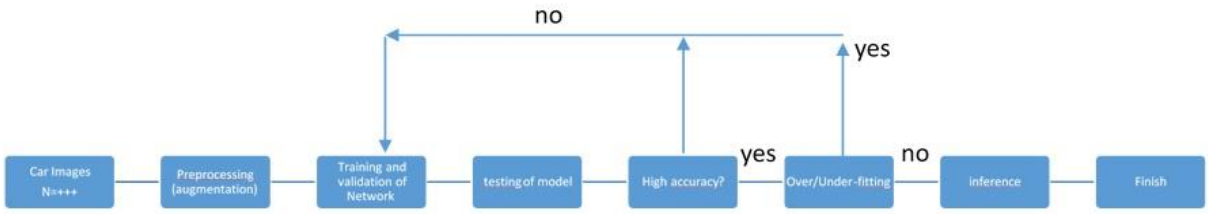

Figure 10: Flow chart of process

It is important to note that we trained three different models: one for each camera angle. The tables and data that follow are valid for one model.
For example, we trained our networks with 265 pictures of cars, this mean that we used 265 pictures of cars per network: i.e.: 265*3 pictures. As you can see in the following tables.

Model	*Number of cars*
Type 1	3*37
Type 2	3*35
Type 3	3*183

Table 1: Number of cars for training per model

State / *Door*	*OPEN*	*CLOSED*
door_front_left	3*67	3*188
door_front_right	3*87	3*168
door_rear_left	3*83	3*172
door_rear_right	3*80	3*175
hood	3*103	3*152
tank	3*39	3*216
trunk	3*39	3*216

Table 2: Training sample of cars

In Table 1and in Table 2 we show the training sample to improve reproductability of our results. If you compare with Table 4 you can see that we have more closed state pictures for training and more opened state pictures for testing. It is because for training we wanted to emphasize the default and correct state to be sure to have this state very well learned by the network. Then for the opened states in the test sample, it is because first of all, there are many different openend states (very small gap, small gap, open) and furthermore we wanted to check all of the different cases, when some opened flaps were hiding others for examples.

To train our three networks pictures of 265 cars per network are used. We then tested the three models using 3*944 pictures of cars in the same proportion as in the 3*265 samples for training. The training/testing set repartition was generated according to the production flow on the days we were on site. The proportion of the types were as follows for the test sample:

Model	*Number of cars*
Type 1	3*179
Type 2	3*121
Type 3	3*644

Table 3: Number of cars for testing per model

To formulate our problem of finding open flaps on a car in a way that a CNN can handle it, we defined 14 different classes for the object detection. In concrete, this means two classes representing the state of open or close for every flap to detect. In Table 4 all classes are shown. To have a sufficient amount of recorded car picture per class, the cars were modified manually during production. Table 4 also shows how many car pictures for each class were considered.
After the acquisition of the pictures, they were labeled by using the software labelme [27]. The classes we used were one class per object and per status: for example, the door_front_left_open and door_front_left_closed are two different objects. These classes are the objects that are searched for in the images.

State / *Door*	*OPEN*	*CLOSED*
door_front_left	3*767	3*177
door_front_right	3*693	3*251
door_rear_left	3*771	3*173
door_rear_right	3*760	3*184
hood	3*373	3*571
tank	3*854	3*90
trunk	3*856	3*88

Table 4: Test sample of cars

When creating the bounding boxes with labelme, a jsonfile format is obtained that is consecutively transformed into PASCALVOCXML and then CSV using python script in order to finally obtain the desired format TFrecord. That is a format specifically for tensorflow [28]. This format is used for all training and validation data (see Figure 11).

Figure 11: Turning images into input data with Object Label

To test our images we ran the algorithm and made tests according to the flow chart in Figure 10. Changes were made if the results were unsatisfying.

An accuracy of over 95% was aspired, because that is approximately the accuracy a human person can achieve on this specific task, i.e.: the overview of the gaps of a car on the conveyor belt or rather to check if a flap is open or not. This is comprehendible, since a very small gap of an open car door is visually hard to identify, even for humans. This precision was obtained after personally spending days on production lines and seeing how many flaps we were able to detect without external assistance. One of the most important things for the training was that the bounding boxes for the labeled ground truth was big enough to detect the gaps of open flaps. As the object detection algorithm is trying to find the best IoU for its anchor boxes, it can lead to bad detection results if the boxes are too small or thin. To keep detections where the algorithm achieves high confidence rates, we used an IoU threshold of 0.6. This corresponds to a confidence rate of 60% for the boxes and can be seen exemplarily in Figure 11. According to [20] the bounding box threshold and parameters for their detection should be high. In the literature an IoU value of 0.2 [20] or 0.5[23] is recommended, but in our evaluations we found many cases where the algorithm was confident with 55% or 58%. Therefore, an IoU of 60% is feasible for us to achieve high quality results.

However, this can also lead to bad results, as smaller gaps were undetected. To avoid this, we took 0.2 as the minimum aspect ratio and 0.2 also as the minimum area for the minimum object covered. In Figure 12 you can see our output.

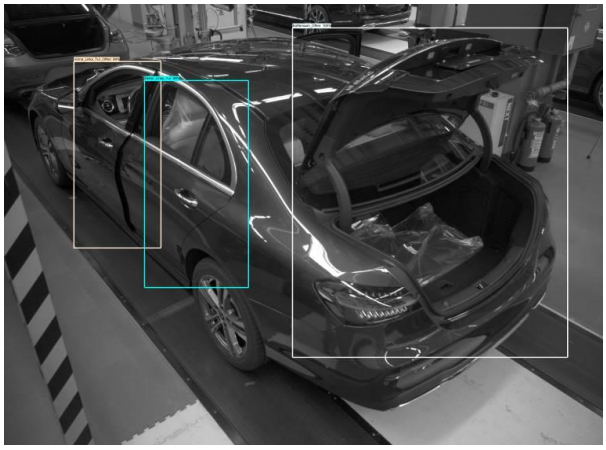

Figure 12: Output of our system. With confidence values of 99%, 99% and 100% from left to right

5. RESULTS

For the CNN training and detection a test system with the computing power of an i7-10750H-2,60GHZ CPU, 32GB RAM and an NVIDIA quatro T2000 graphics card with 4GB Ram was used.

The training time of 265 images for one of the three cameras was 4 hours and 30 minutes with a batch size of two. In Figure 13 the learning rate for 25.000 steps is shown. The networks were parametrized with a decaying learning rate for the training as it is more interesting to slow down the training as the model is converging.

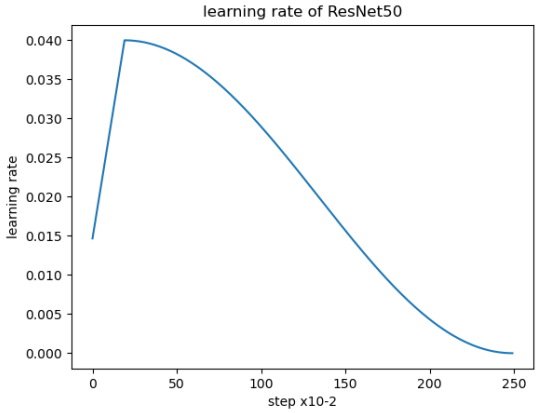

Figure 13: Learning Rate

The losses and training time for each of our three models are similar. That is why we only displayed curves for one model (right side camera). When training the algorithm we recorded the localization and classification losses. These represent the inaccuracy of the predictions and as seen in the sequence (see Figure 14), they continuously diminish in amplitude and value.

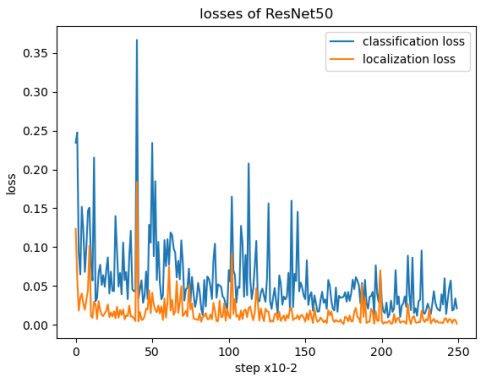

Figure 14: Classification and Localization Loss

The classification loss is the ability of the network to find the appropriate class and the localization loss is the ability to find the class at the right place. In Figure 14 it can be seen that both of these losses are converging towards zero. Even if there is still some noise along this decay, this is a proof for the convergence and performance of our networks. The losses are continuously striving towards zero, meaning that the network is finding more and more the right objects at the right place.

To avoid overfitting and help the networks be less likely prone to make highly nonlinear decisions, an optimization function was introduced that minimizes the global loss with a regularization term. This aims at having weights as close to zero as possible. Having a look at the regularization losses (see Figure 15), it can be clearly seen that the convergence of the losses towards zero shows the diminishing necessity of regularization.

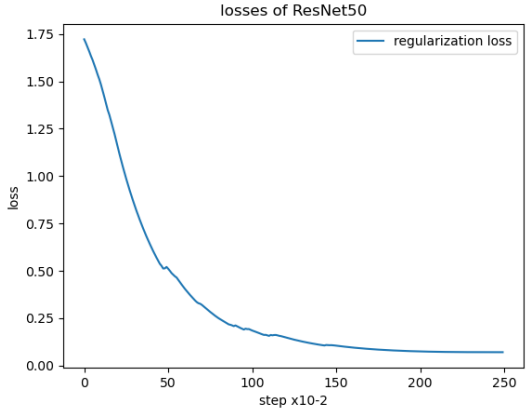

Figure 15: Regularization loss

To measure the performance of our algorithm, the results are presented in a confusion matrix (see Table 5). In object detection a true positive is if the right object is detected at the right place, a false positive is a false detection, a false negative is when the ground truth object is present but the algorithm didn't detect it. A true negative is every part of the picture where no object was predicted. As it is irrelevant in object detection, it can be ignored in the confusion matrix.

After training, 3*944 new car images per camera were used to make predictions and measure the performance of the algorithm. The results are shown in Table 5 with a separate matrix for each camera.

confusion matrix	
True Positive	False Positive
False Negative	True Negative

Front camera	
943	2
1	0

Left camera	
941	4
1	0

Right camera	
939	6
3	0

Table 5: Confusion Matrix

It is important to note that the total of each matrix can be above 944 (number of image used for testing), because if we detect more doors than there are in the test images, it is a false positive.

In Table 6 the results of the three cameras are put together by having a look on the results per car and not per camera.

935	12
5	0

Table 6: Confusion matrix per car in regards to the detections of the 3 cameras together

During setup, we had several complications. During the normal production flow, it is hard to control every factor and it leads to abnormalities.

For example, some people may walk in front of the cameras precisely when the picture is taken. There could also be missing pieces on cars such as bumpers. Finally, there also were papers or additional tape on some cars to point out faults that had to be corrected later on. Even if those faults or errors are very infrequent they still have an impact on the results of the predictions for those cases, where the gaps of the car are occluded. If a paper is in the middle of the hood for example and nothing is occluded the prediction works properly. During normal production flow this shouldn't happen since our setup is a defined field test.

To overcome these errors in the future new classes specifically for these errors can be defined, so that they become properly identified during production flow.

With the results of Table 5 and Table 6 the precision can be calculated, which is the ability to find only relevant objects and also the recall, which is the

ability to find the truth. Precision can be seen as the evaluation of *quality* and recall as the evaluation of *quantity*. The results for the precision and recall can be found in Table 7. They were computed by using the results of Table 6.

	Formula	Value
Precision	TP/(TP+FP)	0.987328405
Recall	TP/(TP+FN)	0.994680851

Table 7: Precision and recall from Table4

With our method a recall rate of 99.5% was achieved, which means that in the majority of the cases the right objects were found. This is very important for our use case of finding open flaps, since our method is able to find the right gaps for 99.5% of the cases. A recall of 100% may be technically reachable under perfect production situations, but in reality if a station is not fenced it may happen that someone walks through the camera image and occludes the car.

The precision is 98.7% and shows the ability of our method to determine only the relevant gaps. In many of the false positive cases the algorithm would have detected a door open with a confidence of 80% and the same door closed with a confidence of 65%. As we take all results above 60% we have a true positive and a false positive at the same time. Even if the door is really open and the algorithm is having more trust in detecting the door open, this leads to a smaller precision. This is the most common error we found in our examinations. To solve this in the future further post processing rules or conditions can be introduced, e.g. that only one object in the class front door (open/close) has to be detected and that always the one with the highest confidence should be taken as the result.

When a human is watching over the production flow a success rate of approximately 95% is achievable because small gaps can be missed because of boredom, tiredness or lacking attention. Looking at the entire vehicle with our method, a sensitivity of 98.7% was achieved (see Table 7), which is superior to the sensitivity of a human by 3.7%.

Finally, we empirically found in the evaluation of our method that these good results are only achievable if the IoU is above 60%.

6. CONCLUSION AND FUTURE WORK

In this paper the new application of deep learning and transfer learning for the industrial use case of detecting open flaps on cars on a conveyor belt was shown. The modeling and break down of the problem into a classification problem was presented, as well as a detailed evaluation of the method and discussion of the challenges. By merely using picture details for the training, the presented method is nearly independent to changes in the factory in the image background. This kind of application and its evaluation is one of the first in the industry and contributes to the literature in this domain of applied research. Very high precision and recall rates over 98 % have been achieved, whereas the errors arised from lacking car parts, occlusions by passing peoples or never seen objects, e.g. paper with checklists, that are added in the picture details used for detection. This is only a problem if the gap between car parts is mostly occluded. The presented method was trained with three different car models and due to the usage of transfer learning, it will be easily adaptable for other models in the future. This is part of our future work. Moreover, this approach can be very useful in other production lines and for the detection of other objects or gaps.

7. REFERENCES

[1] Scheer,F., Neumann,M., Wirth,K., Ginader,M., Oezkurt,Y., Mueller,S.: Evaluation of model-based tracking and its application in a robotic production line. Journal of WSCG, 2020

[2] Scheer,F., Loos, M., Neumann,M.: Model-Based Tracking on Conveyor Belts: Evaluation and Practical Results in the Automotive Industry, WSCG, 2021.

[3] Kang, D. S., Lee, J. W., Ko, K. H., Kim, T. M., Park, K. B., Park, J. R., ... & Lim, D. W., "A study on measurement and compensation of automobile door gap using optical triangulation algorithm." Journal of the Korea Society of Die & Mold Engineering 14.1, 8-14, 2020.

[4] Liu, L., Ouyang, W., Wang, X., Fieguth, P., Chen, J., Liu, X., & Pietikäinen, "Deep learning for generic object detection: A survey." International journal of computer vision 128.2, pp.261-318, 2020.

[5] Derman, E., & Salah, A. A. , "Continuous real-time vehicle driver authentication using convolutional neural network-based face

recognition." 2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018). IEEE, 2018.

[6] Kosmopoulos, D., & Varvarigou, T., "Automated inspection of gaps on the automobile production line through stereo vision and specular reflection." Computers in Industry 46.1, 49-63, 2001.

[7] Mazzetto, M., Teixeira, M., Rodrigues, É. O., & Casanova, D., "Deep learning models for visual inspection on automotive assembling line." arXiv preprint arXiv:2007.01857, 2020.

[8] Stappen, L., Du, X., Karas, V., Müller, S., & Schuller, B. W., "Go-CaRD–Generic, Optical Car Part Recognition and Detection: Collection, Insights, and Applications." arXiv preprint arXiv:2006.08521 (2020).

[9] Rahimi, A., Anvaripour M., and Hayat K. "Object Detection using Deep Learning in a Manufacturing Plant to Improve Manual Inspection." 2021 IEEE International Conference on Prognostics and Health Management (ICPHM). IEEE, 2021.

[10] Malik, H. S., Dwivedi, M., Omakar, S. N., Samal, S. R., Rathi, A., Monis, E. B., ... & Tiwari, A., "Deep Learning Based Car Damage Classification and Detection." EasyChair Preprint 3008,2020.

[11] Kyu, P. M., & Woraratpanya, K.. "Car damage detection and classification." Proceedings of the 11th international conference on advances in information technology, 2020.

[12] Vedang C., Surgenor B. "Fault detection and classification in automated assembly machines using machine vision." The International Journal of Advanced Manufacturing Technology 90.9, 2491-2512, 2017.

[13] Popescu, D., Ichim, L., "Image based fault detection algorithm for flexible industrial assembly line." 22nd International Conference on Control Systems and Computer Science (CSCS). IEEE, 2019.

[14] Behrendt, K., Witt, J., "Deep learning lane marker segmentation from automatically generated labels." IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2017.

[15] Mingxing T.,Le Q.V., "Efficientnet: Improving accuracy and efficiency through AutoML and model scaling." arXiv preprint arXiv:1905.11946, 2019.

[16] Haisong H., Wei Z.,Yao L., "A novel approach to component assembly inspection based on mask R-CNN and support vector machines." Information 10.9, 282, 2019.

[17] deep-learning - Why is resnet faster than vgg - Cross Validated. https://stats.stackexchange.com/questions/280179/why-is-resnet-faster-than-vgg/280338. Accessed 8 november 2021.

[18] Fung, V. "An overview of resnet and its variants." Towards data science, 2017

[19] Reynolds, Anh H. « Anh H. Reynolds ». Anh H. Reynolds, https://anhreynolds.com/. visited 17 mars 2022.

[20] Tsung-Yi Lin, Priya Goyal, Ross B. Girshick, Kaiming He, Piotr Doll: Focal Loss for Dense Object Detection, IEEE Transactions on Pattern Analysis and Machine Intelligence, 2020, https://arxiv.org/pdf/1708.02002.pdf

[21] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun, Deep Residual Learning for Image Recognition, https://arxiv.org/pdf/1512.03385.pdf

[22] Ji, Qingge & Huang, Jie & He, Wenjie & Sun, Yankui. (2019). Optimized Deep Convolutional Neural Networks for Identification of Macular Diseases from Optical Coherence Tomography Images. Algorithms. 12. 51. 10.3390/a12030051.

[23] « Intersection over Union (IoU) for Object Detection ». PyImageSearch, 7 november 2016, https://www.pyimagesearch.com/2016/11/07/intersection-over-union-iou-for-object-detection/.

[24] « Détection d'objets SSD: Détecteur MultiBox Single Shot pour un traitement en temps réel ». ICHI.PRO, https://ichi.pro/fr/detection-d-objets-ssd-detecteur-multibox-single-shot-pour-un-traitement-en-temps-reel-171355751058950. Seen 15 march 2022.

[25] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, Serge Belongie, Feature Pyramid Networks for Object Detection, https://arxiv.org/abs/1612.03144

[26] Bozinovski, S., Fulgosi, A.: The influence of pattern similarity and transfer learning upon the training of a base perceptron B2." Proceedings of Symposium Informatica 1976.

[27] Wada, K. Labelme: Image Polygonal Annotation with Python [Computer software]. https://doi.org/10.5281/zenodo.5711226

[28] Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jozefowicz, R., Jia, Y., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Schuster, M., Monga, R., Moore, S., Murray, D., Olah, C., Shlens, J., Steiner, B.,

Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., & Zheng, X. (2015). TensorFlow, Large-scale machine learning on heterogeneous systems [Computer software]. https://doi.org/10.5281/zenodo.4724125

[29] Coco data set (Common Objects in Context), https://cocodataset.org, visited March 2020.

[30] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, Alexander C. Berg: SSD: Single Shot MultiBox Detector, ECCV 2016, https://arxiv.org/pdf/1512.02325.pdf

.