2019 IEEE Intelligent Transportation Systems Conference (ITSC)
Auckland, NZ, October 27-30, 2019

Towards Robust CNN-based Object Detection through Augmentation with Synthetic Rain Variations

Georg Volk[1], Stefan Müller[1], Alexander von Bernuth[1], Dennis Hospach[1] and Oliver Bringmann[1]

***Abstract*— Convolutional Neural Networks (CNNs) achieve high accuracy in vision-based object detection tasks. For their usage in the automotive domain, CNNs have to be robust against various kinds of natural distortions caused by different weather conditions while state-of-the-art datasets like KITTI lack these challenging scenarios. Our approach automatically identifies corner cases where CNNs fail and improves their robustness by automated augmentation of the training data with synthetic rain variations including falling rain with brightness reduction as well as raindrops on the windshield. Our method achieves higher performance upon validation against a real rain dataset compared with state-of-the art data augmentation techniques like Gaussian noise (GN) or Salt-and-Pepper noise (SPN).**

I. Introduction

Verifying functional safety of fully automated and autonomous vehicles is one of the most important challenges of the coming years in the automotive domain. Such a vehicle has to operate not only under ideal conditions, but also under challenging and unforeseeable situations. Advances in the development of neural networks are one of the key factors to enable autonomous cars. Especially the validation of neural networks remains a serious problem for the car manufacturers. Even with the recommended $10^9\,\mathrm{km}$ for ISO 26262, which an autonomous vehicle should cover for qualification [1], it is almost impossible to cover all kinds of input variations for such neural networks.

Recent news about fatal accidents with autonomously driving vehicles show that these systems are still not reliable under very common conditions like driving on a freeway or at night [2], [3]. Hence, it is important to improve the verification process of such systems to guarantee robust algorithms and enable functional safety. Formal verification of robustness is currently impossible for larger networks [4]. A pixel based semantic verification does not allow any statement about the robustness because one misclassified pixel does not affect functional safety. Hence, the functional safety has to be verified on a more abstract level with meaningful properties [5]. For robustness verification of neural networks it is crucial to cover as many different scenarios and environmental conditions as possible. The collection of this data on the road represents a tough challenge as it is impossible to collect reproducible scenarios containing environmental influences like rain, snow or fog. Furthermore, existing datasets like the KITTI Vision Benchmark Suite [6], Cityscapes [7] or the recently released ApolloScape [8] lack environmental influences and only contain images with good weather conditions. For verification under challenging weather conditions, such conditions must be present in the validation phase of neural networks. Moreover, for a robustness optimization these weather influences have to be considered in the training phase as well. A feasible approach is to use current datasets like KITTI or Cityscapes and synthetically induce weather conditions to evaluate and optimize neural networks. This allows to extend already existing real scenarios with realistic variations of virtual rain to realize an augmented dataset containing all desired conditions. A benefit of these augmentation approaches are comparable and reproducible results.

In this paper we present an approach to extract the functional boundaries of neural networks by applying synthetic rain variations on the KITTI dataset. We first show that our synthetic rain model produces equal effects on object detection algorithms like real rain. The robustness evaluation and optimization of neural networks will be shown on Faster-RCNN [9] and YOLO [10] as an example. The results of the optimization with synthetic rain variations are compared against the optimization with Gaussian noise (GN) and/or Salt-and-Pepper noise (SPN). This comparison will be performed on a real rain dataset, which is described in section VI. Additionally an already robust network, RRC [11], will be compared against the optimized versions of YOLO and Faster-RCNN. The workflow of this approach is illustrated in Figure 1.

II. Related work

Research on CNN-based object detection methods mainly focuses on improving the accuracy. Yet, there are still other important issues to consider before the deployment of such methods in an autonomous vehicle is possible: The runtime of the networks, the size of the network model and especially the robustness of such networks are equally important factors. Neural networks have to operate in all kinds of conditions present on streets. Hence, they have to be robust against environmentally induced noise.

To test the robustness of neural networks against different kinds of noise, Dodge and Karam [12] investigated how noise like blurring and image compression affects state of the art networks. They showed that noisy images have strong effects on classification tasks. Da Costa et al. [13] analyzed the impact of three different noise types (Gaussian-, Poisson- and Salt-and-Pepper noise) and showed how denoising can

[1]University of Tübingen, Faculty of Science, Department of Computer Science, Embedded Systems, {georg.volk, stefan.mueller, alexander.von-bernuth, dennis.hospach oliver.bringmann}@uni-tuebingen.de

*This work has been partially funded by the Deutsche Forschungsgemeinschaft (DFG) in the priority program 1835 under grant BR2321/5-1.

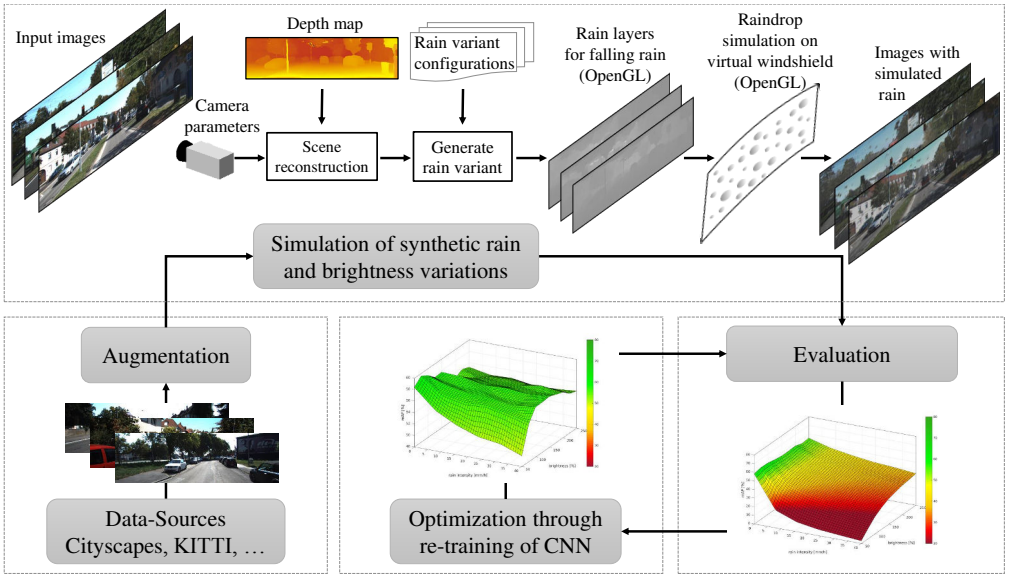


Fig. 1: Workflow of robustness optimization and evaluation by simulating rain variations.

partially compensate the performance drop. Nazaré et al. [14] investigated the influence of disturbances such as GN and SPN on CNNs. Their results have shown that the classification accuracy of CNNs decreases with the introduction of noise to the image to be classified.

In general, the accuracy of CNNs largely depends on the used training dataset and network architecture. To generate additional training data, image transformations are often applied to existing training data. Montserrat et al. investigated various methods to extend the training dataset and thus increase the accuracy of the CNNs in [15]. They presented an approach to extend the training set by applying geometric and color transformations as well as blur and noise to the training data to gain a more robust recognition. The authors in [16], [17] also investigated the influence of disturbances in the input data and tried to improve the accuracy of the CNNs by deliberately including noisy images in the training process. Dresossi et al. introduced a counterexample-guided data augmentation framework [18]. They used a virtual environment to automatically generate scenes and identify counterexamples which are misclassified by the neural network to extend the datasets with these counterexamples. They generated virtual scenes by varying the number of cars, their positions, the image brightness, sharpness and contrast. Zheng et al. [19] focused on the effects of typical image processing tasks such as compression, rescaling and cropping on state of the art Deep Neural Networks (DNNs). They proposed a new network architecture, which is more robust against the aforementioned noise types and automatically considers altered images within the training phase. Most of the approaches in [12]–[14], [16] only investigate generic sources of error such as GN, blur or SPN to evaluate and improve CNNs and do not generate realistic environmental influences, which would occur in reality.

For more realistic disturbances a recent work from Hasirlioglu and Riener [20] modeled effects of rain on camera sensors. They pre-computed a noise filter mask for a camera sensor by distributing water drops within the field of view of the camera. This mask was applied to a static scene and compared against artificial rain with an intensity $100\,\mathrm{mm/h}$ produced by an indoor test system. A weakness of their approach is that depth information is only used for generating the filter mask and neglected within the image to be augmented. This results in one static mask and a big visual difference between artificial and simulated rain. Another approach to achieve more robust object detection was introduced by Mukherjee et al. in [21]. They tried to eliminate rain on images with a CNN trained on images with simulated rain. Their rain model used depth maps generated from monocular images resulting in inaccurate depth information and unrealistic rain.

With DeepTest, Tian et al. [17] presented a methodology to automatically evaluate autonomous driving DNNs with augmented data to detect erroneous behaviors. In their method they considered weather scenarios like rain, fog and lighting conditions to test DNNs for robustness. Nevertheless the induced synthetic rain was not realistically generated. They used just one specific rain model and applied it to all images without taking depth information of the scene into account. Furthermore they lack a method to improve the robustness of the evaluated DNNs. Similarly to DeepTest, DeepXplore by Pei et al. [22] also detected erroneous behavior of neural networks by augmentation of the input images. They induced rectangles onto images or changed lighting conditions. Contrary to Tian et al. they additionally performed an optimization by augmenting training data and achieved an improved detection accuracy.

Another methodology is the usage of Generative Adversarial Networks (GANs) to improve the robustness of neural networks. Luc et al. [23] have shown an application of adversarial training on semantic segmentation to improve labeling accuracy and reduce overfitting. Fischer et al. [24]

presented a first approach to perform adversarial attacks on semantic segmentation. They induced noise with an adversarial attack such that the neural network was unable to detect pedestrians. The network segmented a drivable corridor right through the present pedestrians. Their work showed the need of robust neural networks such that they are able to correctly detect road users even in the presence of noise. Arnab et al. [25] used adversarial attacks to perform a robustness evaluation on neural networks. Karacan et al. [26] used GANs in combination with semantic image information to synthetically generate different environmental conditions. However, the augmentation techniques used in [17], [22], [24]–[26] are not proven to be realistic and lack a comparison against disturbances from real life scenes. A validation using metrics of image processing algorithms to investigate if augmentation techniques produce equal effects as e.g. real rain on machine vision is necessary to allow the use of these augmentation techniques for robustness improvement.

III. Evaluation and Optimization Workflow

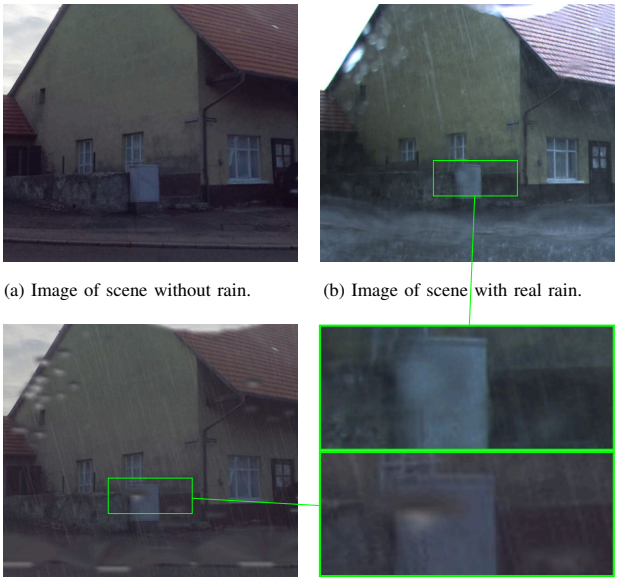
The proposed workflow as illustrated in Figure 1 consists of two stages. After the neural network to be evaluated and optimized has been trained on the initial dataset, the robustness against synthetic rain variations is evaluated. Therefore, the original dataset gets augmented with variations of synthetic rain including brightness reduction, falling rain and raindrops on the windshield. The exact augmentation procedure is explained in section IV-A. In the evaluation phase, defined parameter ranges are investigated which specify the synthetic rain. After identification of critical parameters, the original dataset gets extended in the optimization phase. Images augmented by synthetic rain variations specified by the before identified critical parameters get added. Finally, the neural network gets retrained on the extended dataset.

IV. Robustness Evaluation

The evaluation is based on Faster-RCNN and improved YOLO in version 3 (YOLOv3) [27]. Faster-RCNN is a well known artificial neural network for object detection. It is the basis for many current object detection approaches like YOLO and part of the pedestrian detector in [28]. It consists of two networks, a Region Proposal Network (RPN) followed by the actual Faster-RCNN detector network.

For training the KITTI dataset from Geiger et al. [6] is employed, which supplies labeled traffic scenes. The data was split into two disjoint sets. A training set with 6800 images and a test set with 468 images. The test set contains only 468 images as this set is used to identify critical parameters for later data augmentation. The actual evaluation will be performed on a bigger real world dataset containing heavy rain scenes.

Faster-RCNN was trained on all KITTI labels containing car, pedestrian, person sitting, truck, van, cyclist, tram and misc. For training the 4-step alternating training as presented by Ren et al. in [9] was used. With this training method the RPN and Faster-RCNN network are trained alternately. In the first two steps RPN and Faster-RCNN detector network will be trained after they got initialized with a pre-trained ImageNet model. The Faster-RCNN detector network additionally uses the region proposals from RPN for initialization in step two. In step three and four the shared convolution layers among RPN and the Faster-RCNN detector network will be fix and not updated anymore. Only the RPN unique layers in step three and the specific layers for the detector network in step four will be fine-tuned. Step one and three for RPN were trained for 80k iterations and step two and four for Faster-RCNN were trained for 40k iterations.



(a) Image of scene without rain. (b) Image of scene with real rain.

(c) Image of scene with synthetic rain. (d) Real rain vs. synthetic rain.

Fig. 2: Comparison of our synthetic rain and brightness augmentation technique against real rain.

Like Faster-RCNN, YOLOv3 was trained on all KITTI labels. For training the darknet neural network framework with a binary cross-entropy loss for class prediction was used as proposed in [27]. YOLOv3 originally works on images with a size of 608x608 pixels. We adapted the network to work with images of size 1152x312 pixels so that the proportions fit to the KITTI dataset. We trained YOLOv3 for 40k iterations with a learning rate of 0.00001.

To enable a comparison of our optimization approach against an already robust network we trained RRC [11] on all KITTI labels as well. To train RRC the presented method and parameter configuration from [11] was utilized. For the baseline training of these three neural networks (no augmentation of training data) the same disjoint split in training and test set of the KITTI dataset was used.

A. Data augmentation through variations of synthetic rain

The first step for robustness evaluation is to investigate the influence of data augmentation on the performance of the previously trained neural networks on the mentioned KITTI-based datasets. To reconstruct the 3D scene from left and right images from the KITTI dataset the Slanted Plane Smoothing Stereo algorithm of Yamaguchi et al. [29] is used.

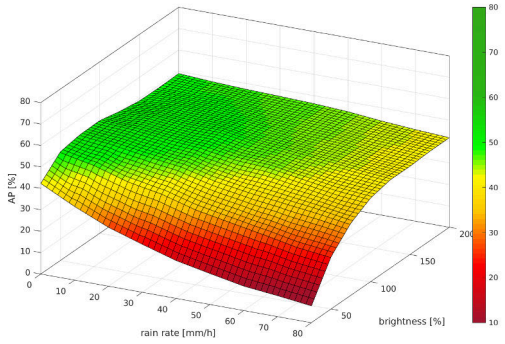
Instead of using standard noise distributions as proposed in [12]–[14], [16] or simply one specific rain filter as in [17], we use variations of realistic synthetic rain to modify the datasets and evaluate the robustness. The augmentation consists of two steps, the generation of falling rain followed by rendering raindrops on the windshield. The methodology for generation of synthetic falling rain on images has already been described in the work of Hospach et al. [30]. In summary, scene depth is used to reconstruct a 3D representation of the input images, then rain streaks are distributed in the space between camera and background respecting physical properties. For the generation of raindrops on the windshield the approach presented by von Bernuth et al. [31] was used. Similar to [30] the scene depth is used to reconstruct the 3D scene. Then raindrops are distributed on the virtual windshield and raytracing is used for a physically correct rendering of these raindrops on the virtual windshield.

We show the visual realism of the synthetic rain model by comparing the same scene without rain (2a), with real rain (2b) and with synthetic rain (2c). For better visibility the same image extract is enlarged in (2d). The real rain image (2b) as well as the synthetic rain (2c) have identical rain streaks, blur effects and drops, showing that the used rain model produces similar optical effects as real rain.

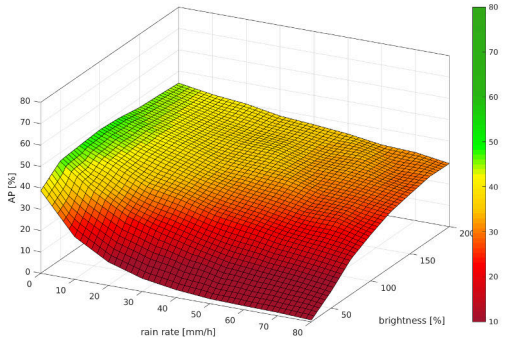
Before we start the evaluation of Faster-RCNN and YOLOv3 we perform a deeper validation of the used rain model for falling rain. After capturing many measurements containing real rain it has been observed that rain has a significant influence on metrics of basic image processing like for example Harris features [30]. Therefore, these basic image processing algorithms applied on artificial rain have been compared to the metrics on real rain images of the same scene. Many complex image processing algorithms are assembled using such basic algorithms. Faster-RCNN for example utilizes edge detectors for feature extraction on the first layers. Hence, edge detection based algorithms (SURF, Canny, Harris, Sobel) allow a deliberate generalization for validating the realism of this rain generation model. The original publication by Hospach et al. [30] already contains validation results of one image set containing real rain scenes. This has now been extended by two additional validation sets of different rain intensities. To acquire the images for validation, the following setup has been used: A series of images of a well-textured scene containing enough edges and corners for algorithms to detect was recorded under real rain (denoted as RefReal). The rain intensity was averaged during acquisition (in this example: cumulated intensity of $52\,\mathrm{mm/h}$, averaged over a time period of 15 minutes). Immediately after rain has stopped, another image sequence of the same scene without rain has been acquired (denoted as RefClean). Using RefClean as input, synthetically simulated rain variations with intensities of $10\,\mathrm{mm/h}$, $40\,\mathrm{mm/h}$, $70\,\mathrm{mm/h}$ and $100\,\mathrm{mm/h}$ (denoted as SimX) have been generated and compared against RefClean and RefReal.

The correct correspondences of the 20 best Harris features of seven randomly chosen frames were compared against a separate reference image of RefClean. The average of correctly identified correspondences in RefReal dropped to 15.71 from originally 16.27 in RefClean. Sim40 was closest to RefReal with an average of 15.57 correct correspondences, while Sim70 has shown a more extreme drop to 13.86 correspondences, respectively 13.14 correspondences for Sim100. A simulation run without rain streaks, denoted as Sim0 has shown that the simulation itself does not generate unwanted side effects during simulation. Its metric value of 16.27 is equal to RefClean.

Other validation results were in close agreement with the presented example, showing that the used model for generating synthetic rain variations produces similar effects compared to real rain. This is of great importance as the following results of this work make use of this assumption.



(a) Robustness of the Faster-RCNN baseline against rain and brightness variations.



(b) Robustness of the YOLOv3 baseline against rain and brightness variations.

Fig. 3: Robustness evaluation for the two neural networks to be optimized.

TABLE I: Detection results of Faster-RCNN network on KITTI validation test set over all augmentation variations.

Network	mean AP	min AP	max AP	std-dev AP	mean AA
Faster-RCNN	41.45%	2.92%	52.67%	10.49%	44.47%
YOLOv3	33.74%	0.19%	48.07%	12.60%	57.26%

B. *Parameter space exploration*

The parameter space to generate scenarios of the critical situations was explored using grid search. In our example, we have six parameters for data augmentation with synthetic rain

TABLE II: Parameter ranges for data augmentation in the evaluation and optimization phase of our workflow.

Case	Parameters	Value intervals
GN	μ	$[10, 50]$
	σ	$[1, 20]$
SPN	density	$[1\%, 30\%]$
GN & SPN	μ	$[10, 50]$
	σ	$[1, 20]$
	density	$[1\%, 30\%]$
Synthetic rain	rain intensity r_i (evaluation)	$[0\,\mathrm{mm/h}, 80\,\mathrm{mm/h}]$
	rain intensity r_i (optimization)	$[30\,\mathrm{mm/h}, 80\,\mathrm{mm/h}]$
	rain angle r_a	$[-30°, 30°]$
	brightness r_b (evaluation)	$[25\%, 200\%]$
	brightness r_b (optimization)	$[40\%, 100\%]$
	drop count d_{count}	$[1000, 2000]$
	mean drop radius d_μ	$[0.3\,\mathrm{mm}, 0.8\,\mathrm{mm}]$
	std dev of drop radius d_σ	$[0.25\,\mathrm{mm}, 1.25\,\mathrm{mm}]$

including raindrops on the windshield in front of the camera sensor. Falling rain is specified by the rain intensity r_i in $\mathrm{mm/h}$, the angle of falling rain r_a, and brightness relative to the original image r_b. The drops on the windshield are specified by the drop count d_{count}, the mean radius of the drops d_μ and the standard deviation of the drop radius d_σ. Larsen et al. [32] showed hourly precipitation extremes up to $40\,\mathrm{mm/h}$ are possible and have already been investigated in Europe in 2018 (e.g. in June 2018 in southwest of Germany). To cover even the most extreme weather situations we used rain intensities from 0 to $80\,\mathrm{mm/h}$ with a step size of $10\,\mathrm{mm/h}$ for robustness evaluation. We varied the angle of the falling rain from -30° to 30° with step size of 15° and the brightness from 25% to 200% with step size of 25%. The number of present drops d_{count} on the windshield were assumed to be independent of the rain intensity. The number depends on the frequency of the windshield wiper. Therefore we chose a total number of drops randomly between 1000 and 2000. This number represents the drop count on the virtual windshield. As the camera is mounted close to the virtual windshield only a subset of these drops will actually be rendered and visible on the image. In our case the camera is mounted $10\,\mathrm{cm}$ behind the virtual windshield and the windshield has an angle of 63°. The mean d_μ and standard deviation d_σ for the drop radii was calculated in relation to the current rain intensity with the following two equations 1 and 2. The relationship between rain intensity and drop size was approximated with the drop size distribution presented in [33, p. 12]. All parameter ranges, which were used for evaluation are shown in Table II.

$$d_\mu = 0.3\,\mathrm{mm} + 0.5\,\mathrm{mm} \cdot \frac{r_i}{\max(r_i)} \tag{1}$$

$$d_\sigma = 0.25\,\mathrm{mm} + 1\,\mathrm{mm} \cdot \frac{r_i}{\max(r_i)} \tag{2}$$

C. Performance evaluation

To evaluate the quality of object detections the well known average precision (AP) metric is used. Other established benchmarks, such as KITTI [6] or COCO [34], use AP as well to evaluate detection algorithms. Therefore, we use this metric to provide comparable results. The AP metric is described by Everingham et al. in [35]. For the AP metric the area under the precision recall curve gets calculated. The intersection over the union of predicted bounding boxes and ground truth is needed to determine recall and precision values. If the overlap of those boxes is greater than the specified threshold the detection counts as true positive (TP). The difference between evaluation in [35] and our method is that we use a strict threshold of 70% for detections to count as TP. Furthermore, we do not distinguish between different classes and use the same threshold for all existing class labels in KITTI. Duplicate detections will count as false positive (FP). To measure the detection quality of the neural networks, we also evaluate the average accuracy (AA) known from [35]. AA denotes the ratio of TP object detections to all by the neural network detected objects including TP, FP and false negative (FN) ones by evaluating $AA = TP/(TP + FP + FN)$.

The evaluation result to identify the influence of synthetic rain variations on the AP of Faster-RCNN is illustrated in Figure 3a and in Figure 3b for YOLOv3 respectively. The determined AP is plotted over different rain intensities and brightness adaptations for the augmented KITTI test set. The rain intensities implicitly include the depending drop radii. The number of drops is not separately plotted as they are independent of the rain intensity and chosen randomly. Detailed numbers are shown in Table I. Faster-RCNN achieved a mean AP of 41.45% while YOLOv3 achieved a mean AP of 33.74% over all rain and brightness variations. We have trained both networks on all provided KITTI labels and not only on the three labels car, pedestrian and cyclist as usually done for the KITTI benchmark. Therefore the AP of 50.42% (Faster-RCNN) and 48.42% (YOLOv3) without augmentation are not to be confused with the online available results. Additionally the online available AP values are given per class and we evaluate the AP over all classes.

A brightness below 100% and an increasing rain intensity significantly lowers the detection quality of the investigated neural network. The most critical situation for Faster-RCNN ($80\,\mathrm{mm/h}$ rain intensity, 0° rain angle, 25% brightness) resulted in a drop by 94.21% in relation to the AP of 50.42% with no rain and brightness augmentation. For YOLOv3, the most critical situation ($80\,\mathrm{mm/h}$ rain intensity, -30° rain angle, 25% brightness) led to a drop by 99.61% compared to the AP of 48.42% with no augmentation. These results show that a robustness optimization of neural networks is necessary. Current state of the art datasets like KITTI lack weather-influenced scenarios to achieve robust networks solely from training on these datasets.

V. Robustness Optimization

To optimize the robustness, we extended the KITTI training set with augmented data and performed the training of Faster-RCNN and YOLOv3 again on this new dataset. This represents the last step in our workflow as illustrated in Figure 1. The KITTI training set was split as before in a training set consisting of 6800 images and a test set

(a) Original image extract from KITTI dataset [6].

(b) Applied synthetic rain ($r_i = 80\,\mathrm{mm\,h^{-1}}$, $r_a = -20°$, $r_b = 75\%$ and $d_{count} = 1985$).

Fig. 4: Synthetic rain augmentation technique on KITTI dataset.

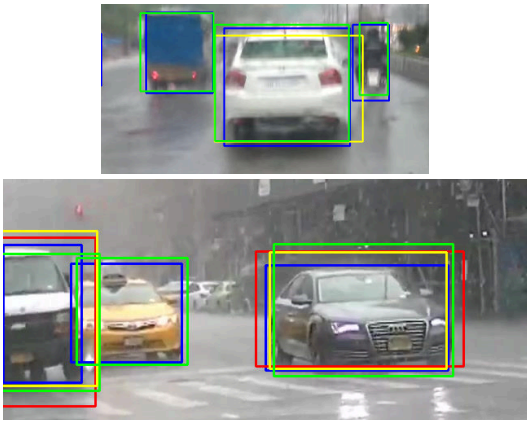
containing 468 images. The whole training set of 6800 images was augmented and added to the original dataset, resulting in a total of 13600 training images. To compare the effectiveness of the data augmentation through synthetic rain and brightness variation, other data augmentation methods were compared against our approach. Therefore the training set was also augmented by applying Gaussian noise (GN), Salt-and-Pepper noise (SPN) and a combination of GN and SPN. A comparison of an original KITTI image vs our synthetic rain augmentation is illustrated in Figure 4.

The training set has been extended with variations of the different augmentations. For synthetic rain six parameters have to be chosen as in the evaluation phase. These parameters were already presented in section IV-B. From the previous evaluation we have identified that only a brightness below 100% affects the neural networks in a negative manner. The rain intensity of the most critical situation was identified with $80\,\mathrm{mm/h}$. Therefore, we constrained the intervals for brightness augmentation and rain intensity to the ranges found as critical in the evaluation phase. Furthermore, we also restricted the lower bound of the brightness augmentation to 40% as this has been shown to be more effective compared to lower brightness values. The lower bound of rain intensity was raised to $30\,\mathrm{mm/h}$, as challenging situations only occurred above this rain intensity.

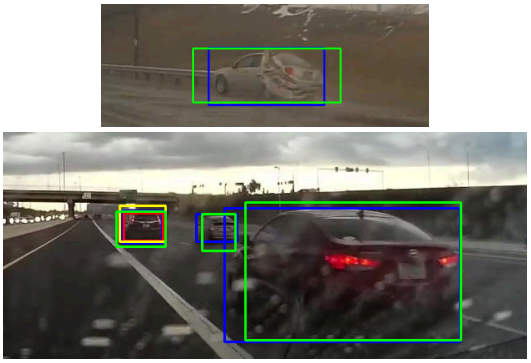
All parameter ranges for generating the augmented training sets are illustrated in Table II. The exact parameter values for every augmentation technique were randomly chosen for each image within the specified parameter ranges to generate a training set of various conditions, except for d_μ and d_σ. The parameters d_μ and d_σ were calculated according to the randomly chosen rain intensity with equations 1 and 2. To be able to reproduce the augmentation the random number generator was seeded. Finally, the initial training set gets extended by the varied training data.

VI. The Realrain Dataset

In order to prove that the optimization with synthetic rain variations also applies under real conditions, a validation on the basis of real scenes is necessary. Existing datasets like KITTI or Cityscapes are lacking necessary weather-influenced scenarios, which are needed for this validation. Hence, we started to collect images of challenging rainy road scenes from our archive of self-conducted test drives and from dashcam videos on YouTube. The result is a dataset of international scenes. In the following, we call it realrain dataset. The realrain dataset contains 2062 labeled images consisting of 9551 labeled objects. It has been subclassified into 7368 cars, 626 vans, 955 trucks, 395 pedestrians, 205 cyclists and one tram. The scenes are well spread from urban to highway scenarios and contain heavy rain, mist and drops on the windshield representing challenging environmental conditions for vision based object detection systems.



(a) Examples containing rain streaks and blur.



(b) Examples containing rain streaks, blur and rain drops on the windshield.

Fig. 5: Comparison of ground truth (blue), Faster-RCNN baseline detection (red), optimization with GN and SPN (yellow) and our optimization with rain and brightness variations (green) on example images taken from our realrain dataset.

VII. Results

The final evaluation of our optimization was performed on the realrain dataset which solely was used for validation. The detection comparison of the baseline, GN and SPN optimization and our optimization on four sample images for Faster-RCNN on the realrain dataset is illustrated in Figure 5. It is quite remarkable that solely our optimized CNN was able to detect with raindrops obstructed vehicles in Figure 5b. The results of these tests are presented in Table III. We can see that our optimization performs best for YOLOv3 as well as for Faster-RCNN considering AP. Second best is GN.

TABLE III: Average precision and accuracy results for Faster-RCNN, YOLOv3 and RRC on the evaluation of our realrain dataset and the original KITTI test set.

		Real Rain Dataset				KITTI Test Set			
Neural Network	Training Method	absolute AP	absolute AA	AP change	AA change	absolute AP	absolute AA	AP change	AA change
Faster-RCNN	Baseline	7.48%	36.22%	0.00%	0.00%	50.42%	41.87%	0.00%	0.00%
	GN	10.20%	38.08%	2.72%	1.86%	**51.02%**	**42.90%**	**0.60%**	**1.03%**
	SPN	7.62%	**39.93%**	0.14%	**3.71%**	48.82%	40.78%	-1.60%	-1.09%
	GN and SPN	9.96%	37.62%	2.48%	1.40%	49.59%	40.98%	-0.83%	-0.89%
	Synthetic rain variations	**11.85%**	34.55%	**4.37%**	-1.67%	49.95%	42.27%	-0.47%	0.40%
YOLOv3	Baseline	5.15%	37.96%	0.00%	0.00%	**48.42%**	**60.93%**	**0.00%**	**0.00%**
	GN	10.30%	42.26%	5.15%	4.30%	45.51%	58.25%	-2.91%	-2.68%
	SPN	3.40%	**43.95%**	-1.75%	**5.99%**	37.72%	59.36%	-10.70%	-1.57%
	GN and SPN	5.10%	42.33%	-0.05%	4.37%	39.61%	56.19%	-8.81%	-4.74%
	Synthetic rain variations	**12.48%**	38.49%	**7.33%**	0.53%	47.79%	59.86%	-0.63%	-1.07%
RRC	Baseline	12.97%	64.33%	-	-	74.60%	74.60%	-	-

We improved the unoptimized detection by 4.37 percentage points for Faster-RCNN and by 7.33 percentage points for YOLOv3. In comparison to the second best optimization with GN we achieved an improvement of 1.65 percentage points for Faster-RCNN and 2.18 for YOLOv3. When we look at the AA of the baseline, it can be seen that the AA decreases by 1.67 percentage points for the optimized Faster-RCNN with synthetic rain variations but gets improved for the optimized YOLOv3 by 0.53 percentage points. The optimization with SPN performs best for the AA metric but also has the worst results considering AP. When it comes to safety under adverse weather conditions not perceiving an obstacle is more severe than false positive detections which e.g. could result in additional breaking maneuvers. Therefore, in our opinion, the AP metric is more relevant than the AA metric for assessing a detectors performance because it considers recall as well as precision.

Furthermore, we compare our two optimized networks to the already robust RRC network. This network is one of the best performing neural networks on the KITTI dataset and enables a good assessment of our results. RRC was ranked seventh place on the KITTI 2D object detection benchmark at the time this paper was written, while YOLOv3 was placed 111th and FasterRCNN 135th. RRC achieves a mean AP of 74.60% on the KITTI testset. This is a lower mean AP value compared to the online available results on the KITTI benchmark website as we trained RRC on all labels and not separately for cars and cyclists together with pedestrians. However, on the realrain dataset RRC only detects 12.97% of the labeled road users correctly showing that even robust networks are incapable of handling adverse weather conditions. Both with rain variations optimized neural networks achieve similar performance like RRC in AP on the realrain dataset, although the unoptimized versions perform drastically worse.

Many data augmentation techniques to enlarge the training set decrease the performance on the original dataset. Therefore we also evaluated the performance of our optimized neural networks on the KITTI test set. The evaluation was performed on the original, not augmented version of the test set to provide an insight into how the data augmentation affects the performance on the original dataset. The results are illustrated in Table III. It can be seen that the optimization with synthetic rain variations almost has no negative effect on the performance on the original KITTI test set. The AP for Faster-RCNN got decreased by 0.47 percentage points and for YOLOv3 by 0.63 percentage points. Looking at the augmentation with GN, the performance of Faster-RCNN increases while it decreases for YOLOv3. The remaining two augmentation techniques including SPN worsen the AP performance slightly for Faster-RCNN but significantly for YOLOv3.

These results show that our approach not only improves the performance of neural networks on the completely different realrain dataset but also maintains the performance on the original dataset. Therefore, we were able to show that the introduction of large variations of synthetic rain into the training process increases the robustness of neural networks against heavy rainfall in real life.

VIII. Conclusion and Outlook

In this paper, we presented a new methodology to evaluate and improve deep neural networks by data augmentation through large variations of synthetic rain. We have shown that current datasets like KITTI are not sufficient to train a robust neural network. The average precision can drop down to 2.92% for Faster-RCNN and 0.19% for YOLOv3 under extreme conditions if the training data does not cover enough scenarios containing bad weather conditions.

Our approach allows for a more precise and focused evaluation as well as training against varying environmental conditions. By means of data augmentation of the KITTI dataset, we achieved an increase of AP by 4.37 percentage points for Faster-RCCN and 7.33 percentage points for YOLOv3 respectively on real rain scenes. This shows the importance of variation in environmental conditions in training data to gain more robust neural networks. Our results look promising and outperform state of the art data augmentations approaches like GN and SPN. This kind of scenario driven augmentation of training data will allow to overcome training weaknesses and enable a targeted evaluation of systems

e.g. for system qualification according to automotive safety integrity level (ASIL).

In the future, we will improve our data augmentation methods by a simulation of additional environmental influences like snow and fog to further increase the robustness. To confirm the generality of our approach we also want to apply the data augmentation of training data to other neural networks. We will also focus our future work on a more sophisticated parameter selection in the data augmentation step to refine the methodology for systematic scenario generation without loss of generality. We would like to investigate how a guided parameter selection with Monte Carlo simulation combined with importance sampling can improve the augmentation. Furthermore, we will try to collect the permissions to publish the final realrain dataset.

REFERENCES

[1] A. Weitzel, H. Winner, C. Peng, S. Geyer, F. Lotz, and M. Sefati, *Absicherungsstrategien fuer Fahrerassistenzsysteme mit Umfeldwahrnehmung*. Fachverlag NW in der Carl Schuenemann Verlag GmbH, 2014.

[2] "Uber Video Shows the Kind of Crash Self-Driving Cars Are Made to Avoid — WIRED." [Online]. Available: https://www.wired.com/story/uber-self-driving-crash-video-arizona/

[3] "Tesla's Self-Driving Autopilot Was Turned On In Deadly California Crash — WIRED." [Online]. Available: https://www.wired.com/story/tesla-autopilot-self-driving-crash-california/

[4] G. Katz, C. Barrett, D. L. Dill, K. Julian, and M. J. Kochenderfer, "Towards Proving the Adversarial Robustness of Deep Neural Networks," in *Proceedings First Workshop on Formal Verification of Autonomous Vehicles*, vol. 257, Turin, Italy, Sep. 2017, pp. 19–26.

[5] S. A. Seshia, A. Desai, T. Dreossi, D. Fremont, S. Ghosh, E. Kim, S. Shivakumar, M. Vazquez-Chanlatte, and X. Yue, "Formal Specification for Deep Neural Networks," Tech. Rep. UCB/EECS-2018-25, 2018.

[6] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset." *I. J. Robotic Res.*, vol. 32, no. 11, pp. 1231–1237, 2013.

[7] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The Cityscapes Dataset for Semantic Urban Scene Understanding," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Las Vegas, NV, USA: IEEE, Jun. 2016, pp. 3213–3223.

[8] X. Huang, X. Cheng, Q. Geng, B. Cao, D. Zhou, P. Wang, Y. Lin, and R. Yang, "The ApolloScape Dataset for Autonomous Driving," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, Jun. 2018.

[9] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 6, pp. 1137–1149, 2016.

[10] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2016, pp. 779–788.

[11] J. Ren, X. Chen, J. Liu, W. Sun, J. Pang, Q. Yan, Y. W. Tai, and L. Xu, "Accurate single stage detector using recurrent rolling convolution," *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, pp. 752–760, 2017.

[12] S. Dodge and L. Karam, "Understanding how image quality affects deep neural networks," in *2016 Eighth International Conference on Quality of Multimedia Experience (QoMEX)*, Jun. 2016, pp. 1–6.

[13] G. B. P. da Costa, W. A. Contato, T. S. Nazare, J. E. S. B. Neto, and M. Ponti, "An empirical study on the effects of different types of noise in image classification tasks," *CoRR*, vol. abs/1609.02781, 2016.

[14] T. S. Nazaré, G. B. P. da Costa, W. A. Contato, and M. Ponti, "Deep Convolutional Neural Networks and Noisy Images," in *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*. Cham: Springer International Publishing, 2018, pp. 416–424.

[15] D. M. Montserrat, Q. Lin, J. Allebach, and E. J. Delp, "Training Object Detection And Recognition CNN Models Using Data Augmentation," in *Electronic Imaging - Imaging and Multimedia Analytics in a Web and Mobile World 2017*, 2017, pp. 27–36.

[16] J. Yim and K.-A. Sohn, "Enhancing the Performance of Convolutional Neural Networks on Quality Degraded Datasets," *Digital Image Computing: Techniques and Applications (DICTA)*, 2017.

[17] Y. Tian, K. Pei, S. Jana, and B. Ray, "DeepTest: Automated Testing of Deep-Neural-Network-driven Autonomous Cars," in *Proceedings of the 40th International Conference on Software Engineering*. Gothenburg, Sweden: ACM, 2018, pp. 303–314.

[18] T. Dreossi, S. Ghosh, X. Yue, K. Keutzer, A. Sangiovanni-Vincentelli, and S. A. Seshia, "Counterexample-Guided Data Augmentation," in *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence*, Stockholm, Sweden, Jul. 2018, pp. 2071–2078.

[19] S. Zheng, Y. Song, T. Leung, and I. Goodfellow, "Improving the Robustness of Deep Neural Networks via Stability Training," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2016, pp. 4480–4488.

[20] S. Hasirlioglu and A. Riener, "A Model-Based Approach to Simulate Rain Effects on Automotive Surround Sensor Data," in *2018 IEEE International Conference on Intelligent Transportation Systems (ITSC)*. Maui, Hawaii, USA: IEEE, Nov. 2018, pp. 2609–2615.

[21] J. Mukherjee, K. Praveen, and V. Madumbu, "Visual Quality Enhancement Of Images Under Adverse Weather Conditions," in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. Maui, Hawaii, USA: IEEE, Nov. 2018, pp. 3059–3066.

[22] K. Pei, Y. Cao, J. Yang, and S. Jana, "DeepXplore: Automated Whitebox Testing of Deep Learning Systems," in *Proceedings of the 26th Symposium on Operating Systems Principles*, ser. SOSP '17. New York, NY, USA: ACM, 2017, pp. 1–18.

[23] P. Luc, C. Couprie, J. Verbeek, and L. J. Kuntzmann, "Semantic Segmentation using Adversarial Networks," *NIPS Workshop on Adversarial Training*, 2016.

[24] V. Fischer, M. C. Kumar, J. H. Metzen, and T. Brox, "Adversarial Examples for Semantic Image Segmentation," in *International Conference on Learning Representations (ICLR) workshop*, Mar. 2017.

[25] A. Arnab, O. Miksik, and P. H. S. Torr, "On the Robustness of Semantic Segmentation Models to Adversarial Attacks," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. Salt Lake City, UT, USA: IEEE, Jun. 2018.

[26] L. Karacan, Z. Akata, A. Erdem, and E. Erdem, "Learning to Generate Images of Outdoor Scenes from Attributes and Semantic Layouts," *CoRR*, vol. abs/1612.00215, Dec. 2016.

[27] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," *CoRR*, vol. abs/1804.02767, 2018.

[28] L. Zhang, L. Lin, X. Liang, and K. He, "Is Faster R-CNN Doing Well for Pedestrian Detection?" in *ECCV 2016*, 2016, pp. 443–457.

[29] K. Yamaguchi, D. McAllester, and R. Urtasun, "Efficient joint segmentation, occlusion labeling, stereo and flow estimation," in *ECCV*, 2014.

[30] D. Hospach, S. Mueller, W. Rosenstiel, and O. Bringmann, "Simulation of Falling Rain for Robustness Testing of Video-Based Surround Sensing Systems," in *Proceedings of the 2016 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2016, pp. 233–236.

[31] A. von Bernuth, G. Volk, and O. Bringmann, "Rendering physically correct raindrops on windshields for robustness verification of camera-based object recognition," in *2018 IEEE Intelligent Vehicles Symposium (IV)*, June 2018, pp. 922–927.

[32] A. N. Larsen, I. B. Gregersen, O. B. Christensen, J. J. Linde, and P. S. Mikkelsen, "Potential future increase in extreme one-hour precipitation events over Europe due to climate change," *Water Science and Technology*, vol. 60, no. 9, pp. 2205–2216, 2009.

[33] M. Roser, "Modellbasierte und positionsgenaue erkennung von regentropfen in bildfolgen zur verbesserung von videobasierten fahrerassistenzfunktionen," Ph.D. dissertation, KIT Scientific Publishing, Karlsruhe, 2012.

[34] T. Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: Common Objects in Context," in *Computer Vision - ECCV 2014*, 2014, pp. 740–755.

[35] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The Pascal Visual Object Classes (VOC) Challenge," *International Journal of Computer Vision*, vol. 88, no. 2, pp. 303–338, Jun. 2010.