Benchmarking Robustness in Object Detection: Autonomous Driving when Winter is Coming

Claudio Michaelis*[a,1,2], Benjamin Mitzkus*[a,1], Robert Geirhos*[a,1,2], Evgenia Rusak*[a,1,2], Oliver Bringmann†[b,1], Alexander S. Ecker†[a,1], Matthias Bethge†[a,1], and Wieland Brendel†[a,1]

[*] *equal contribution*
[†] *joint senior authors*
[a] *first.last@bethgelab.org*
[b] *oliver.bringmann@uni-tuebingen.de*
[1] *University of Tübingen*
[2] *International Max Planck Research School for Intelligent Systems*

Abstract

The ability to detect objects regardless of image distortions or weather conditions is crucial for real-world applications of deep learning like autonomous driving. We here provide an easy-to-use benchmark to assess how object detection models perform when image quality degrades. The three resulting benchmark datasets, termed Pascal-C, Coco-C and Cityscapes-C, contain a large variety of image corruptions. We show that a range of standard object detection models suffer a severe performance loss on corrupted images (down to 30–60% of the original performance). However, a simple data augmentation trick—stylizing the training images—leads to a substantial increase in robustness across corruption type, severity and dataset. We envision our comprehensive benchmark to track future progress towards building robust object detection models. Benchmark, code and data are available at: `http://github.com/bethgelab/robust-detection-benchmark`

Figure 1. Mistaking a dragon for a bird may be dangerous (left) but missing it all together just because it snows (right) is downright suicidal. Sadly, this is exactly the fate that an autonomous agent relying on a state-of-the-art object detection system would suffer. Predictions generated using Faster R-CNN.

Figure 2. Expect the unexpected: To ensure safety, an autonomous vehicle must be able to be recognize objects even in challenging outdoor conditions such as fog, frost, snow, and, of course, the occasional dragonfire.[1]

1 Introduction

> *A day in the near future: Autonomous vehicles are swarming the streets all over the world, tirelessly collecting data. But on this cold November afternoon traffic comes to an abrupt halt as it suddenly begins to snow: winter is coming. Huge snow flakes are falling from the sky, and the cameras of autonomous vehicles are no longer able to make sense of their surroundings, triggering immediate emergency brakes. A day later, an investigation of this traffic disaster reveals that the unexpectedly large size of the snow flakes was the cause of the chaos: While state-of-the-art vision systems had been trained on a variety of common weather types, their training data contained hardly any snow flakes of this size...*

This fictional example highlights the problems that arise when Convolutional Neural Networks (CNNs) encounter settings that were not explicitly part of their training regime. For example, state-of-the-art object detection algorithms such as Faster R-CNN [Ren et al., 2015] fail to recognize objects when snow is added to an image (as shown in Figure 1), even though the objects are still clearly visible to a human eye. At the same time, augmenting the training data with several types of distortions is not a sufficient solution to achieve general robustness against previously unknown corruptions: It has recently been demonstrated that CNNs generalize poorly to novel distortion types, despite being trained on a variety of other distortions [Geirhos et al., 2018]. Even an innocuous distribution shift such as a transition from small snow flakes at training time to large snow flakes at test time can have a strong impact on current vision systems.

On a more general level, CNNs often fail to generalize outside of the training domain or training data distribution. Examples include the failure to generalize to images with uncommon poses of objects [Alcorn et al., 2019] or to cope with small distributional changes [e.g. Zech et al., 2018, Touvron et al., 2019]. One of the most extreme cases are adversarial examples [Szegedy et al., 2013]: images with a domain shift so small that is imperceptible for humans yet sufficient to fool a CNN. We here focus on the less extreme but far more common problem of perceptible image distortions like blurry images, noise or natural distortions like snow.

As an example, autonomous vehicles need to be able to cope with wildly varying outdoor conditions such as fog, frost, snow, sand storms, or falling leaves, just to name a few (as visualized in Figure 2). One of the major reasons why autonomous

[1] Outdoor hazards have been directly linked to increased mortality rates [Lystad and Brown, 2018].

cars have not yet gone mainstream is the inability of their recognition models to function well in adverse weather conditions [Dai and Van Gool, 2018]. Many common environmental conditions can (and have been) modelled, including fog [Sakaridis et al., 2018a], rain [Hospach et al., 2016], snow [von Bernuth et al., 2019] and daytime to nighttime transitions [Dai and Van Gool, 2018]. However it is impossible to foresee all potential conditions that might occur "in the wild".

If we could build models that are robust to every possible image corruption, weather changes would not be an issue. However, in order to assess the robustness of models one first needs to define a measure. While testing models on the set of all possible corruption types is impossible, we argue that a useful approximation is to evaluate models on a diverse range of corruption types that were not part of the training data: if a model copes well with a dozen corruptions that it has never seen before, we expect it to cope well with yet another type of corruption.

In this work, we propose three easy-to-use benchmark datasets termed Pascal-C, Coco-C and Cityscapes-C to assess distortion robustness in object detection. Each dataset contains versions of the original object detection datasets which are corrupted with 15 distortions, each spanning five levels of severity. This approach is directly inspired by Hendrycks and Dietterich [2019], who introduced corrupted versions of commonly used *classification* datasets (ImageNet-C, CIFAR10-C) as standardized benchmarks. After evaluating standard object detection algorithms on these benchmark datasets, we show how a simple data augmentation technique—stylizing the training images—can strongly improve robustness across corruption type, severity and dataset.

1.1 Contributions

Our contributions can be summarized as follows:

1. We demonstrate that a broad range of object detection and instance segmentation models suffer severe performance impairments on corrupted images.
2. To quantify this behaviour and to enable tracking future progress, we propose the `Robust Detection Benchmark`, consisting of three benchmark datasets termed Pascal-C, Coco-C & Cityscapes-C.
3. We show that a simple data augmentation technique—stylizing the training data—leads to large robustness improvements for all evaluated corruptions without any additional labelling costs or architectural changes.
4. We make our benchmark, corruption and stylization code openly available in an easy-to-use fashion:
   - Benchmark,[2] data and data analysis are available at `https://github.com/bethgelab/robust-detection-benchmark`
   - Our pip installable image corruption library is available at `https://github.com/bethgelab/imagecorruptions`
   - Code to stylize arbitrary datasets is provided at `https://github.com/bethgelab/stylize-datasets`

[2]We have integrated evaluation code to assess performance under corruption into the mmdetection toolbox. The code can be found here: `https://github.com/bethgelab/mmdetection` and will be submitted as a pull request to mmdetection.

1.2 Related Work

Benchmarking corruption robustness In recent years, there have been several publications studying the vulnerability of DNNs to common corruptions. Dodge and Karam [2016] measure the performance of four state-of-the-art image recognition models on out-of-distribution data and show that DNNs are in particular vulnerable to blur and Gaussian noise. Geirhos et al. [2018] show that DNN performance drops much faster than human performance for the task of recognizing corrupted images when the perturbation level increases across a broad range of corruption types. Azulay and Weiss [2018] investigate the lack of invariance of several state-of-the-art DNNs to small translations. A benchmark to evaluate the robustness of recognition models against common corruptions was recently introduced by Hendrycks and Dietterich [2019].

Improving corruption robustness One way to restore the performance drop on corrupted data is to preprocess the data in order to remove the corruption. Mukherjee et al. [2018] propose a CNN-based approach to restore image quality of rainy and foggy images. Bahnsen and Moeslund [2018] and Bahnsen et al. [2019] propose algorithms to remove rain from images as a preprocessing step and report a subsequent increase in recognition rate. A challenge for these approaches is that noise removal is currently specific to a certain distortion type and thus does not generalize to other types of distortions. Another line of work seeks to enhance the classifier performance by the means of data augmentation, i.e. by directly including corrupted data into the training. Vasiljevic et al. [2016] study the vulnerability of a classifier to blurred images and enhance the performance on blurred images by fine-tuning on them. Geirhos et al. [2018] study on generalization between different corruption types and find that fine-tuning on one corruption type does not enhance performance on other corruption types. Geirhos et al. [2019] train a recognition model on a stylized version of the ImageNet dataset [Russakovsky et al., 2015], reporting increased general robustness against different corruptions as a result of a stronger bias towards ignoring textures and focusing on object shape. Hendrycks and Dietterich [2019] report several methods leading to enhanced performance on their corruption benchmark: Histogram Equalization, Multiscale Networks, Adversarial Logit Pairing, Feature Aggregating and Larger Networks.

Evaluating robustness to environmental changes in autonomous driving In recent years, weather conditions turned out to be a central limitation for state-of-the art autonomous driving systems [Sakaridis et al., 2018a, Volk et al., 2019, Dai and Van Gool, 2018, Chen et al., 2018, Lee et al., 2018]. While many specific approaches like modelling weather conditions [Sakaridis et al., 2018a,b, Volk et al., 2019, von Bernuth et al., 2019, Hospach et al., 2016, von Bernuth et al., 2018] or collecting real [Wen et al., 2015, Yu et al., 2018, Che et al., 2019, Caesar et al., 2019] and artificial [Gaidon et al., 2016, Ros et al., 2016, Richter et al., 2017, Johnson-Roberson et al., 2017] datasets with varying weather conditions, no general solution towards the problem has emerged. Robustness analysis and optimization of CNNs in the context of autonomous driving is considered by Volk et al. [2019]. Building upon Hospach et al. [2016], Volk et al. [2019] study the fragility of an object detection model against rainy images, identify corner cases where the model fails and include images with synthetic rain variations into the training set.

They report enhanced performance on real rain images. von Bernuth et al. [2018] report a drop in the AP of a Recurrent Rolling Convolution network trained on the KITTI dataset when the camera images are modified by simulated rain drops on the windshield. von Bernuth et al. [2019] model photo-realistic snow and fog conditions to augment real and virtual video streams. They report a significant performance drop of an object detection model when evaluated on corrupted data. Pei et al. [2017] introduce VeriVis, a framework to evaluate the security and robustness of different object recognition models using real-world image corruptions such as brightness, contrast, rotations, smoothing, blurring, and others.

2 Methods

2.1 Robust Detection Benchmark

We introduce the `Robust Detection Benchmark` inspired by the ImageNet-C benchmark for object classification [Hendrycks and Dietterich, 2019] to assess object detection robustness on corrupted images.

Corruption types Following Hendrycks and Dietterich [2019], we provide 15 corruptions on five severity levels each (visualized in Figure 3) to assess the effect of a broad range of different corruption types on object detection models.[3] The corruptions are sorted into four groups: *noise*, *blur*, *digital* and *weather groups* (as defined by Hendrycks and Dietterich [2019]). It is important to note that the corruption types are not meant to be used as a training data augmentation toolbox, but rather to measure a model's robustness against *previously unseen* corruptions. For model validation purposes, the four additional held-out corruption types from ImageNet-C (Speckle Noise, Gaussian Blur, Spatter, Saturate) are provided as well, even though they are not being used to assess performance on the `Robust Detection Benchmark`.

Benchmark datasets The `Robust Detection Benchmark` consists of three benchmark datasets: Pascal-C, Coco-C and Cityscapes-C. Among the vast number of available object detection datasets [Everingham et al., 2010, Geiger et al., 2012, Lin et al., 2014, Cordts et al., 2016, Zhou et al., 2017, Neuhold et al., 2017, Krasin et al., 2017], we chose to use Pascal VOC [Everingham et al., 2010], MS Coco [Lin et al., 2014] and Cityscapes [Cordts et al., 2016] as they are the most commonly used datasets for general object detection (Pascal & Coco) and street scenes (Cityscapes). We follow common convention to select the tests splits (VOC2007 test set for Pascal-C, the Coco 2017 validation set for Coco-C and the Cityscapes validation set for Cityscapes-C).

Performance measures Since performance measures differ between the original datasets, the dataset-specific performance (P) measures are adopted as defined below:

[3]These corruption types were introduced by Hendrycks and Dietterich [2019] and generalized by us to work with arbitrary image dimensions / ratios. Our generalized corruptions can be found at `https://github.com/bethgelab/imagecorruptions` and installed via `pip3 install imagecorruptions`.

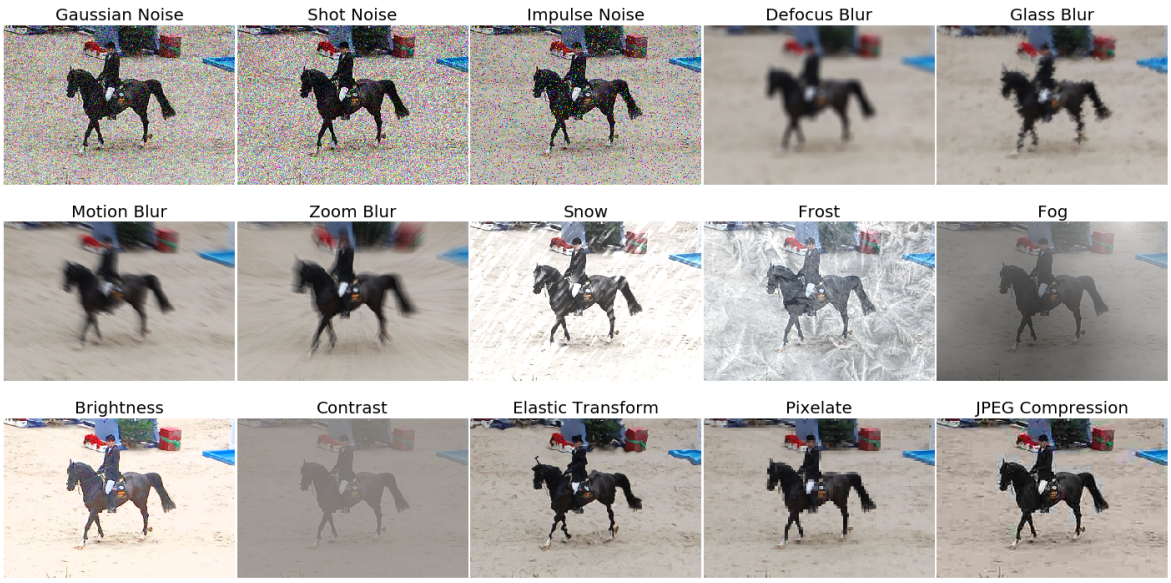


Figure 3. 15 corruption types from Hendrycks and Dietterich [2019], adapted to corrupt arbitrary images (example shown here: randomly selected Pascal VOC image, center crop). Best viewed on screen.

$$\mathrm{P} := \begin{cases} \mathrm{AP}^{50}(\%) & \text{Pascal VOC} \\ \mathrm{AP}(\%) & \text{MS Coco} \\ \mathrm{AP}(\%) & \text{Cityscapes} \end{cases}$$

where AP stands for the 'Average Precision' metric. On the corrupted data, the benchmark performance is measured in terms of mean performance under corruption (mPC):

$$\mathrm{mPC} = \frac{1}{\mathrm{N}_c} \sum_{c=1}^{\mathrm{N}_c} \frac{1}{\mathrm{N}_s} \sum_{s=1}^{\mathrm{N}_s} \mathrm{P}_{c,s} \tag{1}$$

Here, $\mathrm{P}_{c,s}$ is the dataset-specific performance measure evaluated on test data corrupted with corruption c under severity level s while $\mathrm{N}_c = 15$ and $\mathrm{N}_s = 5$ indicate the number of corruptions and severity levels, respectively. In order to measure relative performance degradation under corruption, the relative performance under corruption (rPC) is introduced as defined below:

$$\mathrm{rPC} = \frac{\mathrm{mPC}}{\mathrm{P}_{\mathrm{clean}}} \tag{2}$$

rPC measures the relative degradation of performance on the corrupted data compared to performance on clean data.

Submissions Submissions to the benchmark should be handed in as a simple pull request to the `Robust Detection Benchmark`[4] and need to include all three performance measures: clean performance (P), mean performance under corruption

[4] `https://github.com/bethgelab/robust-detection-benchmark`

Figure 4. Training data visualization for Coco and Stylized-Coco. The three different training settings are: standard data (top row), stylized data (bottom row) and the concatenation of both (termed 'combined' in plots).

(mPC) and relative performance under corruption (rPC). While mPC is the metric used to rank models on the `Robust Detection Benchmark`, the other measures provide additional insights into the causes of a performance gain, as they disentangle gains from higher clean performance (as measured by P) and gains from better generalization performance to corrupted data (as measured by rPC).

Baseline models We provide baseline results for a set of common object detection models including Faster R-CNN [Ren et al., 2015], Mask R-CNN [He et al., 2017], Cascade R-CNN [Cai and Vasconcelos, 2018], Cascade Mask R-CNN [Chen et al., 2019a], RetinaNet [Lin et al., 2017a] and Hybrid Task Cascade [Chen et al., 2019a]. We use a ResNet50 [He et al., 2016] with Feature Pyramid Networks [Lin et al., 2017b] as backbone for all models except for Faster R-CNN where we additionally test ResNet101 [He et al., 2016], ResNeXt101-32x4d [Xie et al., 2017] and ResNeXt-64x4d [Xie et al., 2017] backbones. We additionally provide results for Faster R-CNN and Mask R-CNN models with deformable convolutions [Dai et al., 2017, Zhu et al., 2018] in Appendix Section C. We integrate the robustness benchmark into the `mmdetection toolbox` [Chen et al., 2019b] and train and test all models with standard hyperparameters. The details can be found in Appendix Section A and on the `Robust Detection Benchmark` page.

2.2 Style transfer as data augmentation

For image classification, style transfer (the method of combining the content of an image with the style of another image) has been shown to strongly improve corruption robustness [Geirhos et al., 2019]. We here transfer this method to object detection datasets testing two settings: 1. Replacing each training image with a stylized version. 2. Adding a stylized version of each image to the existing dataset. We apply AdaIN [Huang and Belongie, 2017] with hyperparameter $\alpha = 1$ to the training data, replacing the original texture with the randomly chosen texture information of Kaggle's `Painter by Numbers`[5] dataset. Examples for the stylization of

[5] `https://www.kaggle.com/c/painter-by-numbers/`

Pascal VOC

model	backbone	clean P [AP^{50}]	corrupted mPC [AP^{50}]	relative rPC [%]
Faster	r50	80.5	48.6	60.4

MS Coco

model	backbone	clean P [AP]	corrupted mPC [AP]	relative rPC [%]
Faster	r50	36.3	18.2	50.2
Faster	r101	38.5	20.9	54.2
Faster	x101-32x4d	40.1	22.3	55.5
Faster	x101-64x4d	41.3	23.4	56.6
Mask	r50	37.3	18.7	50.1
Cascade	r50	40.4	20.1	49.7
Cascade Mask	r50	41.2	20.7	50.2
RetinaNet	r50	35.6	17.8	50.1
HTC	x101-64x4d	50.6	32.7	64.7

Cityscapes

model	backbone	clean P [AP]	corrupted mPC [AP]	relative rPC [%]
Faster	r50	36.4	12.2	33.4
Mask	r50	37.5	11.7	31.1

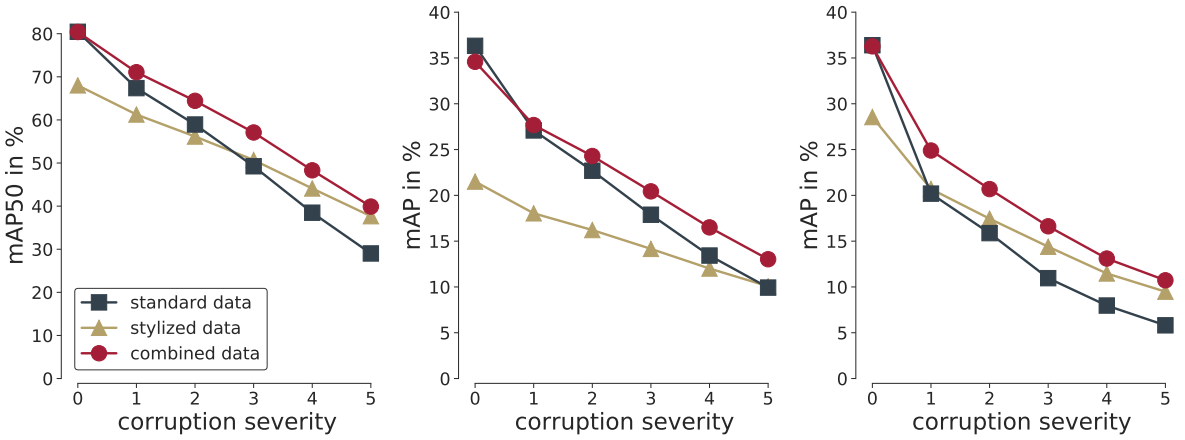
Table 1: Object detection performance of various models. Backbones indicated with r are ResNet and x ResNeXt. All model names exceptfor RetinaNet and HTC indicate the corresponding model from the R-CNN family. All Coco models were downloaded from the mmdetection modelzoo. For all reported quantities: higher is better; square brackets denote metric.

Coco images are given in Figure 4. We provide ready-to-use code for the stylization of arbitrary datasets at `https://github.com/bethgelab/stylize-datasets`.

3 Results

3.1 Image corruptions reduce model performance

In order to assess the effect of image corruptions, we evaluated a set of common object detection models on the three benchmark datasets defined in Section 2. Performance is heavily degraded on corrupted images (compare Table 1). While Faster R-CNN can retain roughly 60% relative performance (rPC) on the rather simple images in Pascal VOC, the same model suffers a dramatic reduction to 33% rPC on the Cityscapes dataset, which contains many small objects. With some variations, this effect is present in all tested models, and also holds for instance segmentation tasks (for instance segmentation results please see Appendix 3).



(a) Pascal-C **(b)** Coco-C **(c)** Cityscapes-C

Figure 5. Object detection robustness of Faster R-CNN on corrupted versions of Pascal VOC, MS Coco and Cityscapes. Corruption severity 0 denotes clean data.

3.2 Robustness increases with backbone capacity

We found corruption resistance to improve with backbone capacity. It seems that almost all corruptions (expect for the blur types) induce a fixed penalty to the encoder, which is not dependent on baseline performance. For two models with different backbones $\Delta\,\mathrm{mPC} \approx \Delta\,\mathrm{P}$ (compare Table 1 and Appendix Figure 10). Therefore, more powerful backbones lead to a relative performance improvement under corruption. This finding is supported when investigating models with deformable convolutions (See Appendix C) and the current state-of-the-art model Hybrid Task Cascade [Chen et al., 2019a], which does not only outperform the strongest baseline model by 9% AP on clean data but distances itself on corrupted data by a similar margin, achieving a leading relative performance under corruption (rPC) of 64.7%. However, not all changes lead to similar improvements. Cascade R-CNN which draws its performance increase from a sophisticated head architecture does not show a relative performance increase to faster R-CNN. This indicates that the improved robustness comes primarily from the image encoding and better head architectures cannot extract more information if the primary encoding is sufficiently impaired.

3.3 Training on stylized data improves robustness

In order to reduce the strong effect of corruptions on model performance observed above, we tested whether a simple approach (stylizing the training data) leads to a robustness improvement. We evaluate the exact same model (Faster R-CNN) with three different training data schemes (visualized in Figure 4):

standard: the unmodified training data of the respective dataset

stylized: the training data is stylized completely

combined: concatenation of standard and stylized training data

The results across our three datasets Pascal-C, Coco-C and Cityscapes-C are visualized in Figure 5. We observe a similar pattern as reported by Geirhos et al.

	Pascal VOC [AP^{50}]			MS Coco [AP]			Cityscapes [AP]		
	clean	corr.	rel.	clean	corr.	rel.	clean	corr.	rel.
train data	P	mPC	rPC [%]	P	mPC	rPC [%]	P	mPC	rPC [%]
standard	**80.5**	48.6	60.4	**36.3**	18.2	50.2	**36.4**	12.2	33.4
stylized	68.0	50.0	**73.5**	21.5	14.1	**65.6**	28.5	14.7	**51.5**
combined	80.4	**56.2**	69.9	34.6	**20.4**	58.9	36.3	**17.2**	47.4

Table 2: Object detection performance of Faster R-CNN trained on standard images, stylized images and the combination of both evaluated on standard test sets (test 2007 for Pascal VOC; val 2017 for MS Coco, val for Cityscapes); higher is better.

[2019] for object classification on ImageNet—a model trained on stylized data suffers less from corruptions than the model trained only on the original "clean" data. However, its performance in on clean data is much lower. Combining stylized and clean data seems to achieve the best of both worlds: high performance on clean data as well as strongly improved performance under corruption. From the results in Table 2, it can be seen that both stylized and combined training improve the relative performance under corruption (rPC). Combined training yields the highest absolute performance under corruption (mPC) for all three datasets. This pattern is consistent for each corruption type and severity level with only a hand full exceptions. Detailed results across corruption types are reported in the Appendix (Figure 7, Figure 8 and Figure 9).

3.4 Performance degradation does not simply scale with perturbation size

We investigated whether there is a direct relationship between the impact of a corruption on the pixel values of an image and the impact of a corruption on model performance. Figure 6 shows the relative performance of Faster R-CNN on the corruptions in Pascal-C dependent on the perturbation size measured in Root Mean Square Error (RMSE). It can be seen that there is no such simple relation. For instance, `impulse noise` alters only few pixels but has drastic impact on the performance of the model, while `brightness` or `fog` alter all pixel values but have small impact onto the model performance. However, there seems to be a relation between the impact of a corruption on model performance and corruption group (*noise*, *blur*, *digital* or *weather*). For instance, the digital corruptions seem to have much lower impact on the performance compared to blur corruptions.

4 Discussion

We here showed that object detection and instance segmentation models suffer severe performance impairments on corrupted images, a pattern that has previously been observed in image recognition models [e.g. Geirhos et al., 2018, Hendrycks and Dietterich, 2019]. In order to track future progress on this important issue, we proposed the `robust detection benchmark` containing three easy-to-use benchmark datasets Pascal-C, Coco-C and Cityscapes-C. Apart from providing baselines against which future models and techniques can be compared, we then demon-

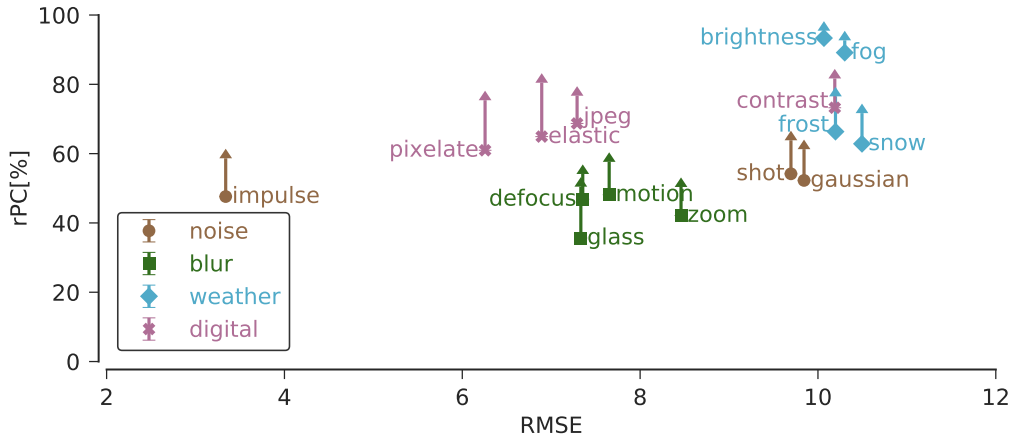


Figure 6. Relative performance under corruption (rPC) as a function of corruption RMSE evaluated on Pascal VOC. The dots indicate the rPC of Faster R-CNN trained on standard data, the arrows show the performance gained via training on 'combined' data. Corruptions are grouped into four corruption types: noise, blur, weather and digital.

strated how a simple data augmentation technique (adding a stylized copy of the training data in order to reduce a model's focus on textural information) leads to strong robustness improvements. On corrupted images, we consistently observe a performance increase (roughly $10-40\%$) with small losses on clean data ($0-2\%$). This approach has the benefit that it can be applied to any image dataset, requires no additional labelling or model tuning, and thus comes basically for free. At the same time, our benchmark data show that there is still space for improvement, and it is yet to be determined whether the most promising robustness enhancement techniques will require architectural modifications, data augmentation schemes, modifications to the loss function, or a combination of these.

We encourage readers to expand the benchmark with novel corruption types. In order to achieve robust models, testing against a wide variety of different image corruptions is necessary, there is no 'too much'. Since our benchmark is open source, we welcome new corruption types and look forward to your pull requests to `https://github.com/bethgelab/imagecorruptions`!

We envision our comprehensive benchmark to track future progress towards building robust object detection models that can be reliably deployed "in the wild", eventually enabling them to cope with unexpected weather changes, corruptions of all kinds, and, if necessary, even the occasional dragonfire.

Author contributions

The initial project idea for improving detection robustness was developed by E.R., R.G. and C.M. The initial idea of benchmarking detection robustness was developed by C.M., B.M., R.G., E.R. & W.B. The overall research focus on robustness was collaboratively developed in the Bethge, Bringmann and Wichmann labs. The `Robust Detection Benchmark` was jointly designed by C.M., B.M., R.G. & E.R.; including selecting datasets, corruptions, metrics and models. B.M. and E.R. jointly developed the pip-installable package to corrupt arbitrary images. B.M. developed code to stylize arbitrary datasets with input from

R.G. and C.M.; C.M. and B.M. developed code to evaluate the robustness of arbitrary object detection models. B.M. prototyped the core experiments; C.M. ran the reported experiments. The results were jointly analysed and visualized by C.M., R.G. and B.M. with input from E.R., M.B. and W.B.; C.M., B.M., R.G. & E.R. worked towards making our work reproducible, i.e. making data, code and benchmark openly accessible and (hopefully) user-friendly. Senior support, funding acquisition and infrastructure were provided by O.B., A.S.E., M.B. and W.B. The illustratory figures were designed by E.R., C.M. and R.G. with input from B.M. and W.B. The paper was jointly written by R.G., C.M., E.R. and B.M. with input from all other authors.

Acknowledgement

We would like to thank Alexander von Bernuth for help with Figure 1; Marissa Weis for help with the Cityscapes dataset; Andreas Geiger for helpful discussions on the topic of autonomous driving in bad weather; Mackenzie Mathis for helpful contributions to the stylization code as well as Eshed Ohn-Bar and Jan Lause for pointing us to important references. R.G. would like to acknowledge Felix Wichmann for senior support, funding acquisition and providing infrastructure. C.M., R.G. and E.R. graciously acknowledge support by the International Max Planck Research School for Intelligent Systems (IMPRS-IS) and the Iron Bank of Braavos. C.M. was supported by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) via grant EC 479/1-1 to A.S.E.. A.S.E., M.B. and W.B. acknowledge support from the BMBF competence center for machine learning (FKZ 01IS18039A) and the Collaborative Research Center Robust Vision (DFG Projektnummer 276693517 – SFB 1233: Robust Vision). M.B. acknowledges support by the Centre for Integrative Neuroscience Tübingen (EXC 307). O.B. and E.R. have been partially supported by the Deutsche Forschungsgemeinschaft (DFG) in the priority program 1835 "Cooperatively Interacting Automobiles" under grant BR2321/5-1 and BR2321/5-2. A.S.E., M.B. and W.B. were supported by the Intelligence Advanced Research Projects Activity (IARPA) via Department of Interior / Interior Business Center (DoI/IBC) contract number D16PC00003.

References

Reidar P Lystad and Benjamin T Brown. "Death is certain, the time is not": mortality and survival in Game of Thrones. *Injury epidemiology*, 5(1):44, 2018.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *NIPS*, 2015.

Robert Geirhos, Carlos RM Temme, Jonas Rauber, Heiko H Schütt, Matthias Bethge, and Felix A Wichmann. Generalisation in humans and deep neural networks. In *NeurIPS*, 2018.

Michael A Alcorn, Qi Li, Zhitao Gong, Chengfei Wang, Long Mai, Wei-Shinn Ku, and Anh Nguyen. Strike (with) a pose: Neural networks are easily fooled by strange poses of familiar objects. In *CVPR*, 2019.

John R Zech, Marcus A Badgeley, Manway Liu, Anthony B Costa, Joseph J Titano, and Eric Karl Oermann. Variable generalization performance of a deep learning

model to detect pneumonia in chest radiographs: A cross-sectional study. *PLoS medicine*, 15(11):e1002683, 2018.

Hugo Touvron, Andrea Vedaldi, Matthijs Douze, and Hervé Jégou. Fixing the train-test resolution discrepancy. *arXiv:1906.06423*, 2019.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv:1312.6199*, 2013.

Dengxin Dai and Luc Van Gool. Dark model adaptation: Semantic image segmentation from daytime to nighttime. In *ITSC*, 2018.

Christos Sakaridis, Dengxin Dai, and Luc Van Gool. Semantic foggy scene understanding with synthetic data. *IJCV*, 2018a.

Dennis Hospach, Stefan Müller, Wolfgang Rosenstiel, and Oliver Bringmann. Simulating photo-realistic snow and fog on existing images for enhanced CNN training and evaluation. In *DATE*, 2016.

Alexander von Bernuth, Georg Volk, and Oliver Bringmann. Simulating photo-realistic snow and fog on existing images for enhanced CNN training and evaluation. In *ITSC*, 2019.

Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. In *ICLR*, 2019.

Samuel Fuller Dodge and Lina J. Karam. Understanding how image quality affects deep neural networks. *QoMEX*, 2016.

Aharon Azulay and Yair Weiss. Why do deep convolutional networks generalize so poorly to small image transformations? *arXiv:1805.12177*, 2018.

Jashojit Mukherjee, K Praveen, and Venugopala Madumbu. Visual quality enhancement of images under adverse weather conditions. In *ITSC*, 2018.

Chris H. Bahnsen and Thomas B. Moeslund. Rain removal in traffic surveillance: Does it matter? *arXiv:1810.12574*, 2018.

Chris H. Bahnsen, David Vázquez, Antonio M. López, and Thomas B. Moeslund. Learning to remove rain in traffic surveillance by using synthetic data. In *VISIGRAPP*, 2019.

Igor Vasiljevic, Ayan Chakrabarti, and Gregory Shakhnarovich. Examining the impact of blur on recognition by convolutional networks. *arXiv:1611.05760*, 2016.

Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A. Wichmann, and Wieland Brendel. Imagenet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness. In *ICLR*, 2019.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015.

Georg Volk, Stefan Müller, Alexander von Bernuth, Dennis Hospach, and Oliver Bringmann. Towards robust CNN-based object detection through augmentation with synthetic rain variations. In *ITSC*, 2019.

Yuhua Chen, Wen Li, Christos Sakaridis, Dengxin Dai, and Luc Van Gool. Domain adaptive faster R-CNN for object detection in the wild. In *CVPR*, 2018.

Unghui Lee, Jiwon Jung, Seokwoo Jung, and David Hyunchul Shim. Development of a self-driving car that can handle the adverse weather. *International journal of automotive technology*, 2018.

Christos Sakaridis, Dengxin Dai, Simon Hecker, and Luc Van Gool. Model adaptation with synthetic and real data for semantic dense foggy scene understanding. In *ECCV*, 2018b.

Alexander von Bernuth, Georg Volk, and Oliver Bringmann. Rendering physically correct raindrops on windshields for robustness verification of camera-based object recognition. *Intelligent Vehicles Symposium (IV)*, pages 922–927, 2018.

Longyin Wen, Dawei Du, Zhaowei Cai, Zhen Lei, Ming-Ching Chang, Honggang Qi, Jongwoo Lim, Ming-Hsuan Yang, and Siwei Lyu. UA-DETRAC: A new benchmark and protocol for multi-object detection and tracking. *arXiv:1511.04136*, 2015.

Fisher Yu, Wenqi Xian, Yingying Chen, Fangchen Liu, Mike Liao, Vashisht Madhavan, and Trevor Darrell. Bdd100k: A diverse driving video database with scalable annotation tooling. *arXiv:1805.04687*, 2018.

Zhengping Che, Guangyu Li, Tracy Li, Bo Jiang, Xuefeng Shi, Xinsheng Zhang, Ying Lu, Guobin Wu, Yan Liu, and Jieping Ye. D2-city: A large-scale dashcam video dataset of diverse traffic scenarios. *arXiv:1904.01975*, 2019.

Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. *arXiv:1903.11027*, 2019.

Adrien Gaidon, Qiao Wang, Yohann Cabon, and Eleonora Vig. Virtual worlds as proxy for multi-object tracking analysis. In *CVPR*, 2016.

German Ros, Laura Sellart, Joanna Materzynska, David Vazquez, and Antonio M. Lopez. The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In *CVPR*, 2016.

Stephan R. Richter, Zeeshan Hayder, and Vladlen Koltun. Playing for benchmarks. In *ICCV*, 2017.

M. Johnson-Roberson, Charles Barto, Rounak Mehta, Sharath Nittur Sridhar, Karl Rosaen, and Ram Vasudevan. Driving in the matrix: Can virtual worlds replace human-generated annotations for real world tasks? In *ICRA*, 2017.

Kexin Pei, Yinzhi Cao, Junfeng Yang, and Suman Jana. Towards practical verification of machine learning: The case of computer vision systems. *arXiv:1712.01785*, 2017.

Mark Everingham, Luc Van Gool, Christopher K. I. Williams, John Winn, and Andrew Zisserman. The Pascal Visual Object Classes (VOC) Challenge. *International Journal of Computer Vision*, 2010.

Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? The KITTI vision benchmark suite. In *CVPR*, 2012.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: Common Objects in Context. In *ECCV*, 2014.

Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *CVPR*, 2016.

Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ADE20K dataset. In *CVPR*, 2017.

Gerhard Neuhold, Tobias Ollmann, Samuel Rota Bulò, and Peter Kontschieder. The mapillary vistas dataset for semantic understanding of street scenes. In *ICCV*, 2017.

Ivan Krasin, Tom Duerig, Neil Alldrin, Vittorio Ferrari, Sami Abu-El-Haija, Alina Kuznetsova, Hassan Rom, Jasper Uijlings, Stefan Popov, Shahab Kamali, Matteo Malloci, Jordi Pont-Tuset, Andreas Veit, Serge Belongie, Victor Gomes, Abhinav Gupta, Chen Sun, Gal Chechik, David Cai, Zheyun Feng, Dhyanesh Narayanan, and Kevin Murphy. Openimages: A public dataset for large-scale multi-label and multi-class image classification. *Dataset available from https://storage.googleapis.com/openimages/web/index.html*, 2017.

Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask R-CNN. In *ICCV*, 2017.

Zhaowei Cai and Nuno Vasconcelos. Cascade R-CNN: Delving into high quality object detection. In *CVPR*, 2018.

Kai Chen, Jiangmiao Pang, Jiaqi Wang, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jianping Shi, Wanli Ouyang, Chen Change Loy, and Dahua Lin. Hybrid task cascade for instance segmentation. In *CVPR*, 2019a.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal Loss for Dense Object Detection. *ICCV*, 2017a.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature Pyramid Networks for Object Detection. In *CVPR*, 2017b.

Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *CVPR*, 2017.

Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *ICCV*, 2017.

Xizhou Zhu, Han Hu, Stephen Lin, and Jifeng Dai. Deformable convnets v2: More deformable, better results. *arXiv:1811.11168*, 2018.

Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, et al. Mmdetection: Open mmlab detection toolbox and benchmark. *arXiv:1906.07155*, 2019b.

Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *ICCV*, pages 1501–1510, 2017.

Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. Accurate, large minibatch SGD: Training ImageNet in 1 hour. *arXiv:1706.02677*, 2017.

Appendix

A Implementation details

Training We train all our models with 2 images per GPU which corresponds to a batch size of 16 on 8 GPUs. On Coco, we resize images so their short edge is 800 pixels and train for 12 epochs with a starting learning rate of 0.02 which is decreased by a factor of 10 after 8 and 11 epochs. On pascal voc, images are resized so that their short edge is 600 pixels. Training is done for 12 epochs with a starting learning rate of 0.01 with a decay step of factor 10 after 9 epochs. For cityscapes, we stayed as close as possible the procedure described in [He et al., 2017] rescaling images to a shorter edge size between 800 and 1024 pixels and train for 64 epochs (to match 24k steps at a batch size of 8) with an initial learning rate of 0.02 and a decay step of factor 10 after 48 epochs. For evaluation, only one scale - 1024 pixels - is used. In all our experiments, we employ the linear scaling rule Goyal et al. [2017] to reduce the learning rate when less than 8 GPUs are used for training. Specifically, we used 4 GPUs to train the Coco models and 1 GPU for all other models resulting in an effective learning rate reduction by a factor of 2 for Coco and 8 for pascal and cityscapes. Training with stylized data is done by simply exchanging the dataset folder or adding it to the list of dataset folders to consider. For all further details please refer to the config files in our implementation.

B Corrupt arbitrary images

In the original corruption benchmark of ImageNet-C [Hendrycks and Dietterich, 2019], two technical aspects are hard-coded: The image-dimensions and the number of channels. To allow for different data sets with different image dimensions, several corruption functions are defined independently of each other, such as `make_cifar_c`, `make_tinyimagenet_c`, `make_imagenet_c`, and `make_imagenet_c_inception`. Additionally, many corruptions expect quadratic images. We have modified the code to resolve these constraints, and now all corruptions can be applied to non-quadratic images with varying sizes, which is a necessary prerequisite for adapting the corruption benchmark to the Pascal VOC and Coco datasets. For the corruption type 'frost', crops from provided images of frost are added to the input images. Since images in Pascal VOC and Coco have arbitrarily large dimensions, we resize the frost images to fit the largest input image dimension if necessary.

The original corruption benchmark also expects RGB images. Our code now allows for grayscale images[6].

Both `motion_blur` and `snow` relied on the motion-blur functionality of Imagemagick, resulting in an external dependency that could not be resolved by standard python package managers. For convenience, we reimplemented the motion-blur functionality in python and removed the dependency on non-python software.

[6]There are approximately 2–3% grayscale images in Pascal VOC/MS Coco.

MS Coco

model	backbone	clean P [AP]	corr. mPC [AP]	rel. rPC [%]
Mask	r50	34.2	16.8	49.1
Cascade Mask	r50	35.7	17.6	49.3
HTC	x101-64x4d	43.8	28.1	64.0

Cityscapes

model	backbone	clean P [AP]	corr. mPC [AP]	rel. rPC [%]
Mask	r50	32.7	10.0	30.5

Table 3: Instance segmentation performance of various models. Backbones indicated with r: ResNet. All model names indicate the corresponding model from the R-CNN family. All models were downloaded from the mmdetection modelzoo.

	MS Coco			Cityscapes		
	clean	corr.	rel.	clean	corr.	rel.
train data	[P]	[mPC]	[rPC]	[P]	[mPC]	[rPC]
standard	**34.2**	16.9	49.4	**32.7**	10.0	30.5
stylized	20.5	13.2	**64.1**	23.0	11.3	**49.2**
combined	32.9	**19.0**	57.7	32.1	**14.9**	46.3

Table 4: Instance segmentation performance of Mask R-CNN trained on standard images, stylized images and the combination of both evaluated on standard test sets (test 2007 for Pascal VOC; val 2017 for MS Coco, val for Cityscapes).

C Additional Results

C.1 Instance Segmentation Results

We evaluated Mask R-CNN and Cascade Mask R-CNN on instance segmentation. The results are very similar to those on the object detection task with a slightly lower relative performance (1%, see Table 3). We also trained Mask R-CNN on the stylized datasets finding again very similar trends for the instance segmentation task as for the object detection task (Table 4). On the one hand, this is not very surprising as Mask R-CNN and Faster R-CNN are very similar. On the other hand, the contours of objects can change due to the stylization process, which would expectedly lead to poor segmentation performance when training only on stylized images. We do not see such an effect but rather find the instance segmentation performance of Mask R-CNN to mirror the object detection performance of Faster R-CNN when trained on stylized images.

MS Coco				
model	backbone	clean P [AP]	corr. mPC [AP]	rel. rPC [%]
Faster	r50-dcn	40.0	22.4	56.1
Faster	x101-64x4d-dcn	43.4	26.7	61.6
Mask	r50-dcn	41.1	23.3	56.7

Table 5: Object detection performance of models with deformable convolutions Dai et al. [2017]. Backbones indicated with r are ResNet, the addition dcn signifies deformable convolutions in stages c3-c5. All model names indicate the corresponding model from the R-CNN family. All models were downloaded from the mmdetection modelzoo.

MS Coco				
model	backbone	clean P [AP]	corr. mPC [AP]	rel. rPC [%]
Mask	r50-dcn	37.2	20.7	55.7

Table 6: Instance segmentation performance of Mask R-CNN with deformable convolutions [Dai et al., 2017]. The backbone indicated with r is a ResNet 50, the addition dcn signifies deformable convolutions in stages c3-c5. The model was downloaded from the mmdetection modelzoo.

C.2 Deformable Convolutional Networks

We tested the effect of deformable convolutions [Dai et al., 2017, Zhu et al., 2018] on corruption robustness. Deformable convolutions are a modification of the backbone architecture exchanging some standard convolutions with convolutions that have adaptive filters in the last stages of the encoder. It has been shown that deformable convolutions can help on a range of tasks like object detection and instance segmentation. This is the case here too as networks with deformable convolutions do not only perform better on clean but also on corrupted images improving relative performance by 6-7% compared to the baselines with standard backbones (See Tables 5 and 6). The effect appears to be the same as for other backbone modifications such as using deeper architectures (See Section 3 in the main paper).

Image rights & attribution

Figure 1: Home Box Office, Inc. (HBO).

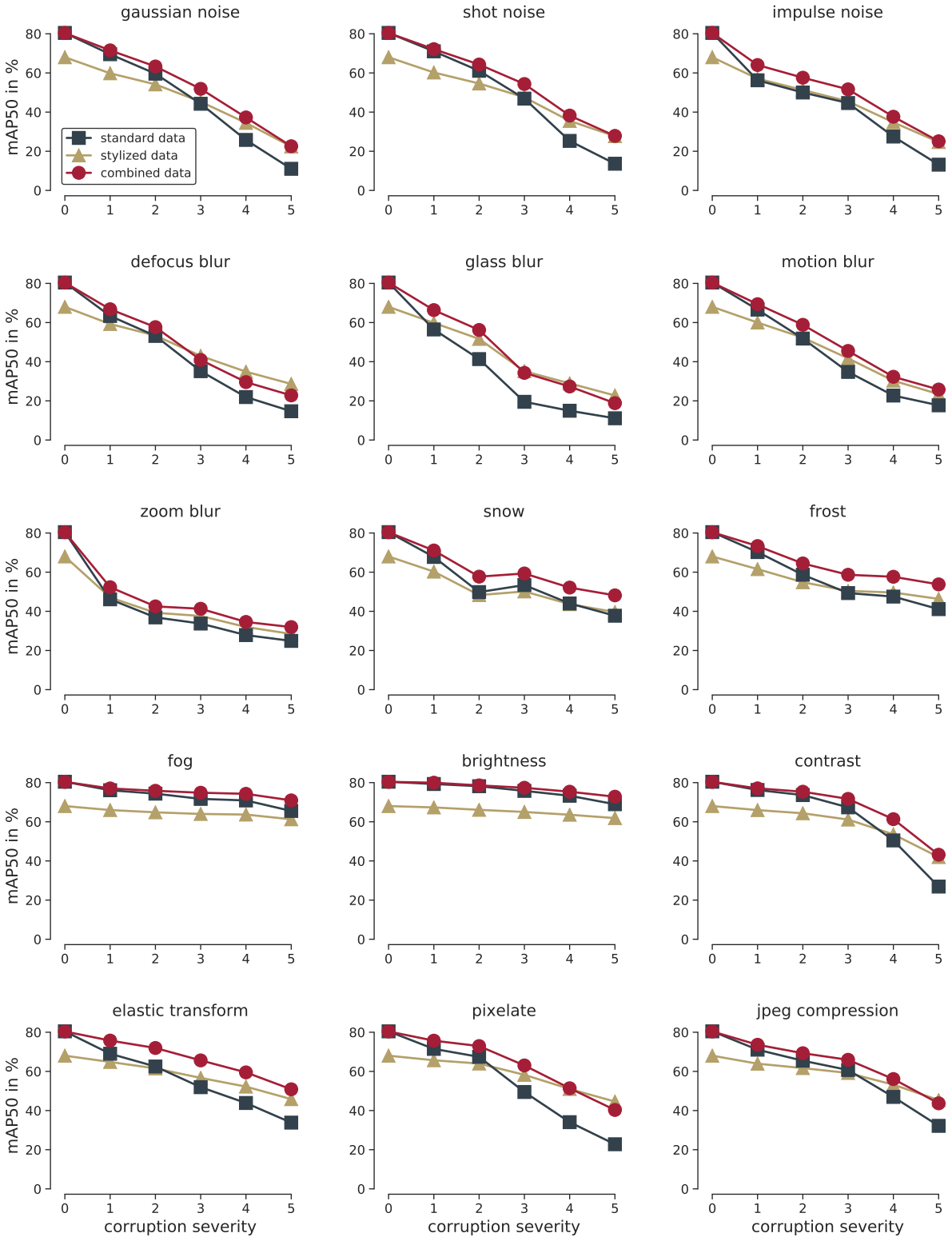


Figure 7. Results for each corruption type on Pascal VOC.

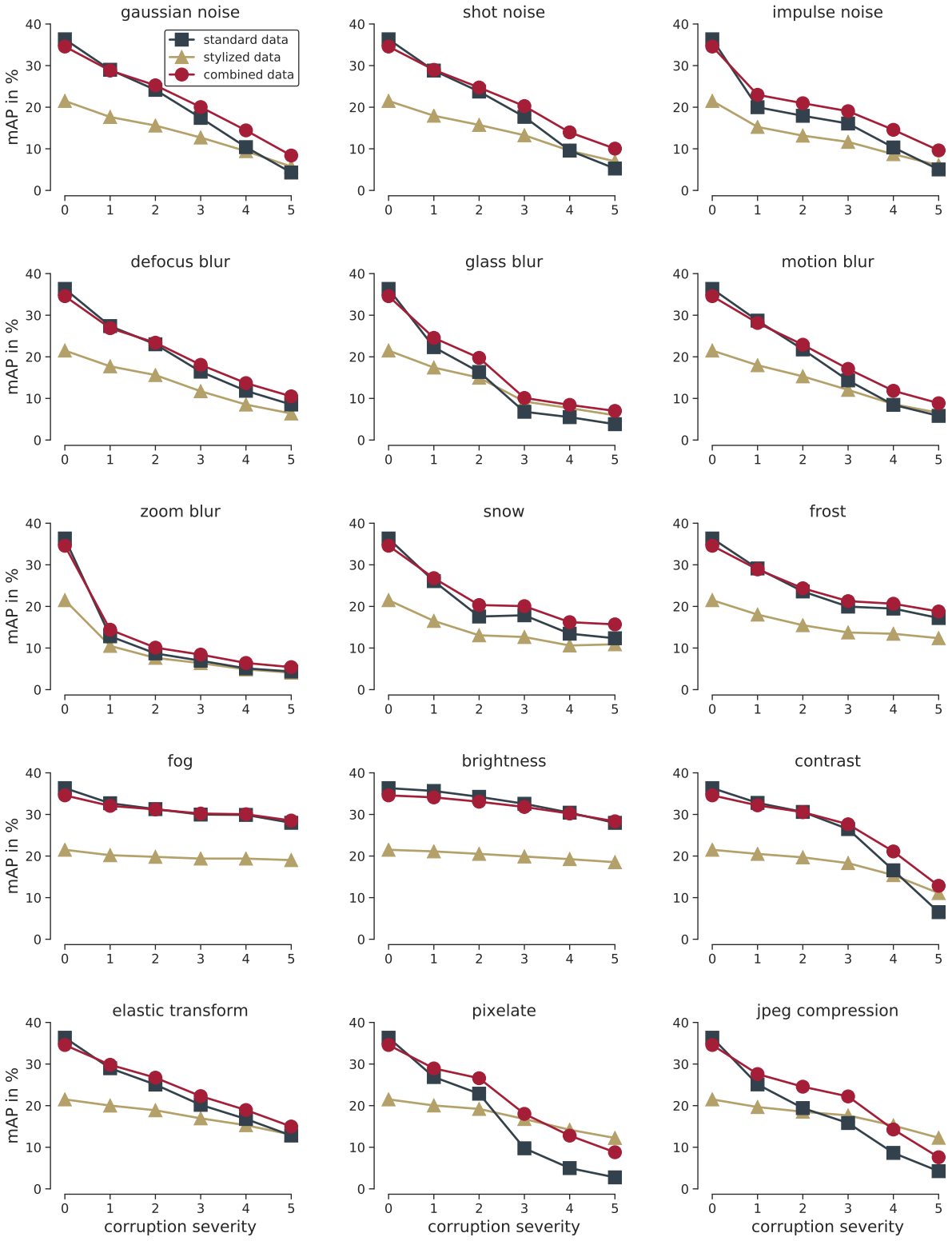


Figure 8. Results for each corruption type on MS Coco.

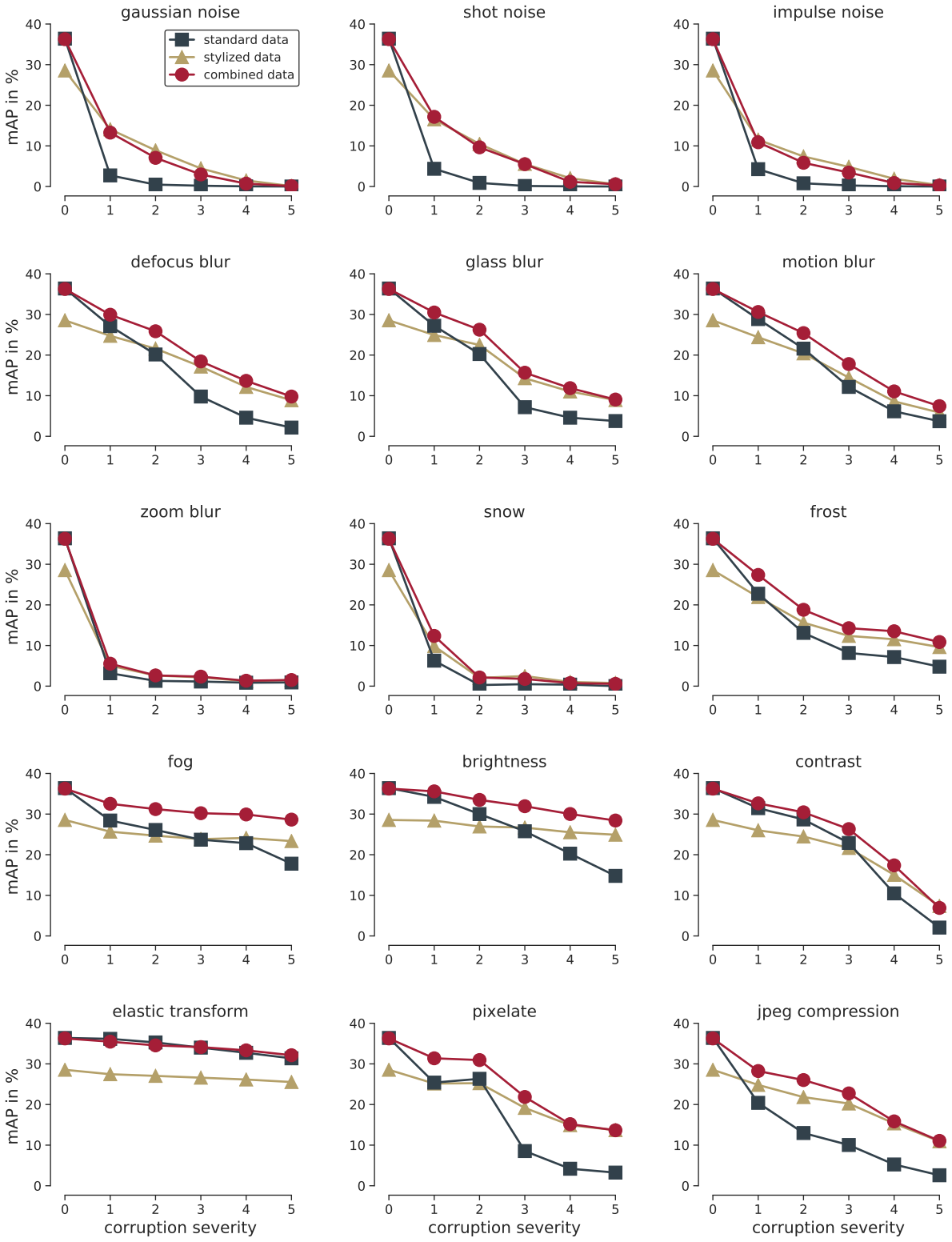


Figure 9. Results for each corruption type on Cityscapes.

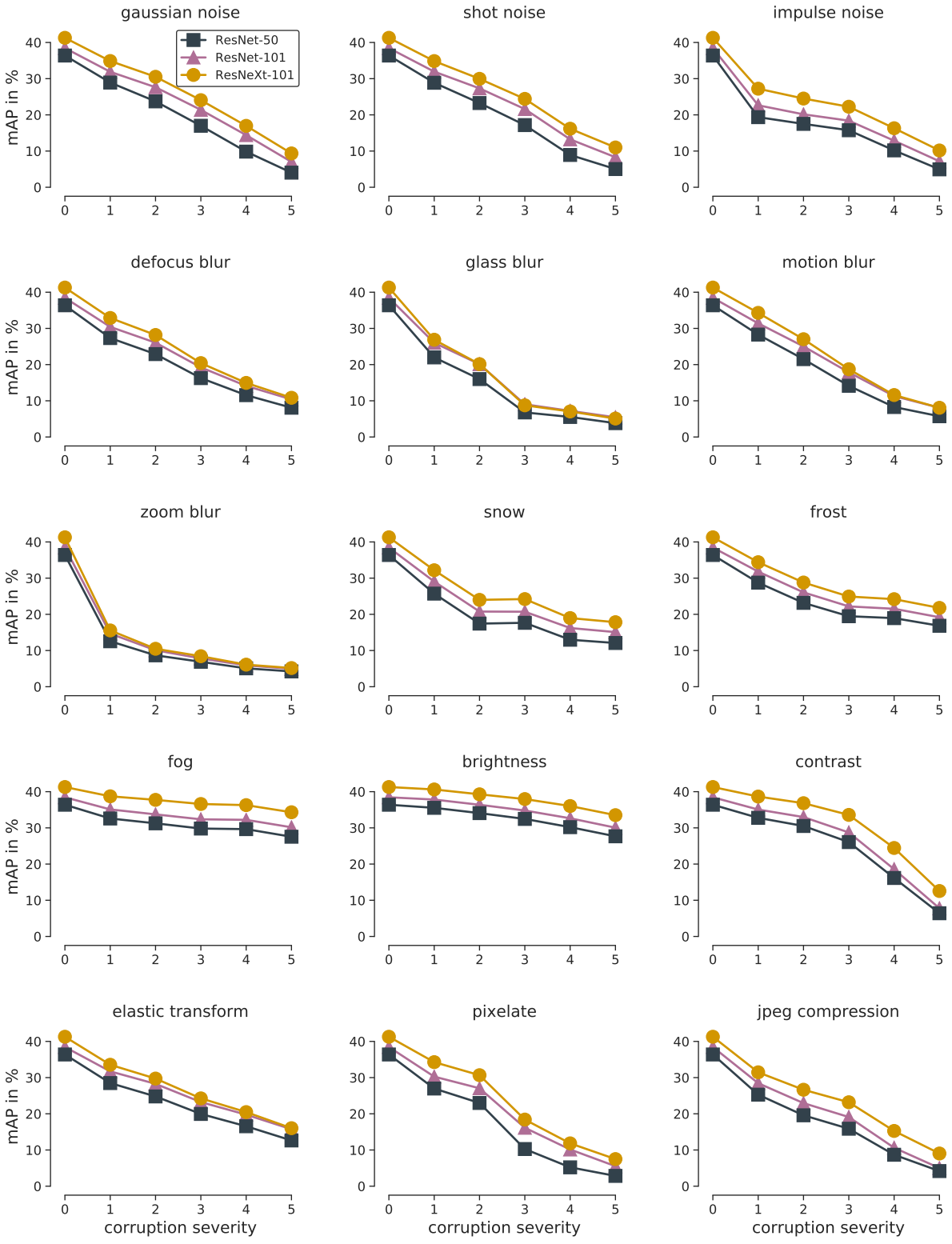


Figure 10. Results for each corruption type using different backbones. Faster R-CNN trained on MS Coco with ResNet-50, ResNet-101 and ResNext-101_64x4d backbones.