Classification robustness to common optical aberrations

Patrick Müller[1,2], Alexander Braun[1], Margret Keuper[2,3]
[1]Hochschule Düsseldorf - University of Applied Sciences,
[2]University of Siegen, [3]Max-Planck-Institute for Informatics, Saarland Informatics Campus

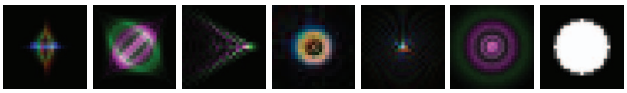
Abstract

Computer vision using deep neural networks (DNNs) has brought about seminal changes in people's lives. Applications range from automotive, face recognition in the security industry, to industrial process monitoring. In some cases, DNNs infer even in safety-critical situations. Therefore, for practical applications, DNNs have to behave in a robust way to disturbances such as noise, pixelation, or blur. Blur directly impacts the performance of DNNs, which are often approximated as a disk-shaped kernel to model defocus. However, optics suggests that there are different kernel shapes depending on wavelength and location caused by optical aberrations. In practice, as the optical quality of a lens decreases, such aberrations increase. This paper proposes OpticsBench*, a benchmark for investigating robustness to realistic, practically relevant optical blur effects. Each corruption represents an optical aberration (coma, astigmatism, spherical, trefoil) derived from Zernike Polynomials. Experiments on ImageNet show that for a variety of different pre-trained DNNs, the performance varies strongly compared to disk-shaped kernels, indicating the necessity of considering realistic image degradations. In addition, we show on ImageNet-100 with* OpticsAugment *that robustness can be increased by using optical kernels as data augmentation. Compared to a conventionally trained ResNeXt50, training with OpticsAugment achieves an average performance gain of 21.7% points on OpticsBench and 6.8% points on 2D common corruptions.*

1. Introduction

Deep neural networks (DNN) are present in peoples' daily lives in various applications to mobility [1, 2, 3], language and speech processing [4] and vision [5, 6].

Although, for example, image classification networks produce excellent results on various challenges, it often remains unclear how well the trained models will generalize from the training distribution to practical test scenarios, that may present them with various domain shifts. For practical deployment, this generalization ability and robustness to data degradation is however crucial. Measuring robustness is addressed by various benchmarks introducing targeted corruptions [7, 8, 9], common corruptions and adverse weather conditions [1, 10, 11] to vision datasets. These methods and benchmarks already cover a wide range of corruptions and yet use necessary simplifications.



(a) (b) (c) (d) (e) (f) (g)

Figure 1: Kernel samples available with OpticsBench (a-f) and a disk-like blur kernel (g). All kernels model optical aberrations, while the kernels in (a-f) additionally include chromatic aberration, asymmetric and non-constant shapes.

In this paper, we focus on a practically highly relevant case that has so far not been addressed in the literature: robustness to realistic optical aberration effects. Under high cost pressure, *e.g.* in automotive mass production on cameras, compromises in optical quality may have to be made and natural production tolerances occur. In practice, even the constant use of a camera can lead to a change in image quality during the period of use, *e.g.* due to thermal expansion. This can lead to increased aberrations [12], *e.g.* chromatic aberration and astigmatism, which are not considered in current robustness research.

This paper proposes OpticsBench to close this gap, a benchmark that includes common types of optical aberrations such as coma, astigmatism and spherical aberration. The optical kernels are derived theoretically from an expansion of the wavefront into Zernike polynomials, which can be mixed by the user to create any *realistic* optical kernel. The dataset grounds on 3D kernels (x, y, color) matched in size to the defocus blur corruption of [10] as we consider the disk-shaped kernel as the base type of blur kernels. We evaluate 70 different DNNs on a total of 1M images from the benchmark applied to ImageNet. Our evaluation shows that the performance of ImageNet models varies strongly for different optical degradations and that the disk-shaped blur kernels provide only weak proxies to estimate the mod-

els' robustness when confronted with optical degradations.

Further, since our analysis indicates a lack of robustness to optical degradations, we propose an efficient tool to use optical kernels for data augmentation during DNN training, which we denote *OpticsAugment*. In experiments on ImageNet-100, OpticsAugment achieves on average 18% performance gain compared to conventionally trained DNNs on OpticsBench. We also show that OpticsAugment allows to improve on 2D common corruptions [10] on average by 5.3% points on ImageNet-100, *i.e.* the learned robustness transfers to other domain shifts.

2. Related work

Vasiljevic et al. [13] investigate the robustness of CNNs to defocus and camera shake. Hendrycks et al. [10] provide a benchmark to 2D common corruptions. The benchmark includes several general modifications to images such as change in brightness or contrast as well as weather influences such as fog, frost and snow. They also include different types of blur, but only consider luminance kernels and more general types such as Gaussian or disk-shaped kernels. Kar et al. [11] build on this work and extend common corruptions to 3D. These include *extrinsic* camera parameter changes such as field of view changes or translation and rotation. Michaelis et al. [1] and Dong et al. [14] provide robustness benchmarks for object detection on common vision datasets. As the field of research grows, more subtle changes in image quality potentially posing a distribution shift are uncovered. This work aims to build on a more general treatment of blur types, which are known in optics research, but less common in computer vision.

A related field of research investigates robustness to adversarial examples created by targeted [15, 16] and untargeted [17] attacks. The goal is to introduce small perturbations of the input data in a way such that the model makes wrong classifications. Successful attacks pose a security risk to a particular DNN, while human observers would not even notice a difference and safely classify [7]. Croce et al. [9] provide a robustness benchmark, originally intended for adversarial robustness testing using AutoAttack [15], while more practical l_p-bounds are discussed in [18]. However, these methods are model specific in that the particular attack is *optimized* for the model using *e.g.* projected gradient descent in the backward pass. Therefore white-box methods require full knowledge about the underlying model. In contrast, model evaluation with OpticsBench corruptions can be done by applying simple filters to the validation data and thus requires only a clean image dataset. Therefore, it also works for black-box models. Convolving tensors with kernels is a base task in computer vision and so GPU-optimized implementations exist. These include parallel evaluation of OpticsBench for many models and on-the-fly training with OpticsAugment.

Since these benchmarks reveal potential distribution shifts or lack of robustness for a given neural network model, concurrently methods are researched to improve robustness [19, 20, 21] on various benchmarks. Hendrycks et al. [22, 23] propose different data augmentation methods to improve classification robustness towards 2D common corruptions. The work of Saikia [24] further builds on this and achieves high accuracy on both clean and corrupted samples. Similarly, methods exist to improve adversarial robustness [25] on benchmarks like [26, 27].

Modeling or retrieving Zernike polynomials [28, 29, 30] is a common process in ophthalmological optics [31, 32] to investigate aberrations of the human eye. The expansion is also widely used in other areas of optics, such as lens design [33], microscopy [34] and astronomy [35, 36]. Therefore several tools [33, 37, 38] exist to generate PSFs from Zernike coefficients or wavefronts. The design is application specific and intended for optical engineers or physicists such as optical design software like Zemax Optic Studio [33]. Our OpticsBench and OpticsAugment leverage such optical models to evaluate and improve current models with respect to realistic optical effects.

3. Blur kernel generation

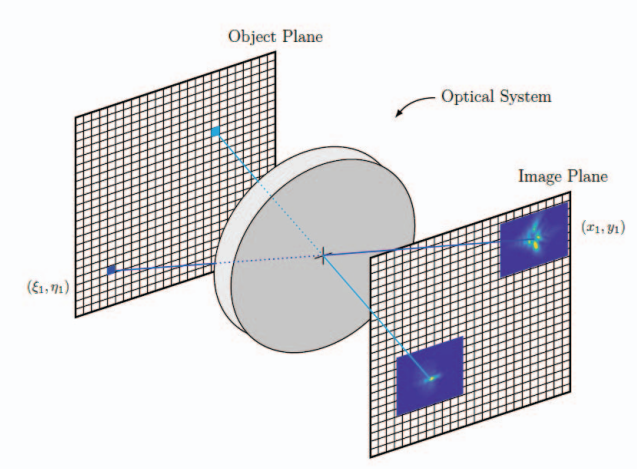


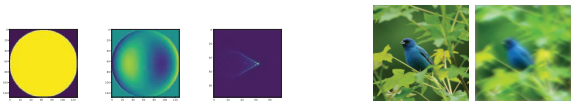
Figure 2: Imaging with space-variant PSF: Point sources in object space that pass the optical system appear spread and differently shaped in image space depending on the angle of incident. Kernel sizes are exaggerated for display purposes.

The point spread function (PSF) is the linear system response to a spherical point wave source emanating at some point in object space. With an ideal lens, this spherical wave would simply converge at the focus. Real-world systems create a shape specific to the aberrations present at that field position. Typically the wavelength-dependent PSF varies from the center to the edge and may also depend on azimuth as shown in Fig. 2. Point sources from different locations

lead to different PSFs [28, 39]. This directly impacts the imaged object: depending on the azimuth, the object appears differently blurred. In this article, we follow a similar approach to show possible effects of lens blur on the robustness of DNNs. For small images of size 224×224 we assume a constant PSF and treat them as if they were regions of interest (ROI) from a larger image: Each image is blurred in different severities and optical aberrations. Since the images already contain different corruptions, including blur, the low pass filtering further reduces the image quality. In addition, depth-dependent blur would also control the blur size at a specific pixel. However, if the objects are beyond the hyperfocal distance, the depth-dependent blur variation can be neglected. This hyperfocal distance is for automotive lenses *e.g.* at several meters. Farther objects have the same blur as objects at *infinity*. Blur is more pronounced for these objects than for near objects because they are much smaller.

3.1. Kernel generation

To obtain kernel shapes specific to real optical aberrations, we use the linear system model for diffraction and aberration as in [39, 28] and expand the wavefront in Zernike polynomials. The effect of a non-ideal lens on a point source of wavelength λ can be compactly summarized by a Fourier transform $\mathcal{F}$ over a circular region that propagates the aberrated wave from pupil space to the image space. Fig. 3 visualizes the process. An ideal spherical wave passes through a circular pupil, which then deforms according to the characteristics of the optical system. If there were no aberrations, the phase would be flat. The



(a) (b) (c) (d) (e)

Figure 3: Image processing scheme: The circular pupil (a) contains an aberrated phase (b), which is mapped into image space by a 2D Fourier Transform, yielding a PSF (c). This PSF is convolved with an image (d) to produce a blurred image (e). The process is repeated for each color channel.

complex phase factor describing the optical path difference W_λ is transformed to image space at z_i to yield a PSF for wavelength λ: [39]

$$h(u, v, \lambda) = |\mathcal{F}\{(Circ(x, y) \cdot e^{-j\frac{2\pi}{\lambda z_i}\mathbf{W}_\lambda(x,y,\lambda)}\}|^2. \quad (1)$$

A blurred image is then created with a 2D convolution of the PSF and the image. The model from Eq. (1) assumes scalar diffraction theory and therefore no polarization. Since we use an l_1-normed discrete PSF to preserve intensity in the final image, any scalar weights are suppressed. In this paper, we consider a very small imager $224 \times 224 \times 3$ that can be fed with ImageNet data. The model is restricted to represent the lens with a single PSF, no dependence on the object distance and angle, no magnification to concentrate on the kernel's shape nor its possible variation over the field of view. We observe the PSF h in a region $u, v \in [0, 25)$ pixels and apply it as a convolution kernel to the images. These restrictions allow for fast processing although an extension to space-variant optical models is given in [40, 41, 42, 43].

Table 1: First twelve Zernike Fringe modes. We select numbers 4-11 for OpticsBench. However, an extension to other (higher) modes is easily accomplished.

piston #	name	#	name
1	piston	**7**	**horizontal coma**
2	tilt x	**8**	**vertical coma**
3	tilt y	**9**	**spherical**
4	**defocus**	**10**	**oblique trefoil**
5	**oblique astigmatism**	**11**	**vertical trefoil**
6	**vertical astigmatism**	12	sec. vert. astigmatism

Now, to model specific PSFs, the expansion of the wavefront W_λ into a complete and orthogonal set of polynomials Z_n^m named after Fritz Zernike [28, 30, 44] is used:

$$W_\lambda(x, y, \lambda) = \lambda \cdot \sum_{n,m} A_n^m(\lambda) \cdot Z_n^m \quad (2)$$

Each coefficient A_n^m in multiples of the wavelengths λ_i represents the contribution of a particular type of aberration and therefore different aspects such as the amount of coma, astigmatism or defocus can be turned off or on. In this article, we choose the eight isolated Zernike Fringe Polynomials, including primary and more complex aberrations. Tilt x and tilt y are not considered here. The concrete list is marked as bold text in Tab. 1. Conversely to Seidel aberrations not occurring isolated but mixed in practice, we select isolated Zernike modes, combining Seidel aberrations by definition, such as astigmatism & defocus. For convenience, we refer to these aberrations here briefly as their Seidel aberration equivalent. However, lens aberrations of real systems usually consist of a bunch of different Zernike modes, see e.g. the Zemax sample objectives, we select here single Zernike modes to allow for categorization and encourage researchers to combine different modes and investigate its impact on model robustness. Although we do not show distortion, tilt, and field curvature here, the same framework can be used to model or process entire lens representations. This is particularly useful for image datasets with larger images, as the lens corruptions then may vary significantly over the field of view, *e.g.* the Berkeley Deep Drive (BDD100k). [2] The PSF generation is done in Python and the pupil modeling is based on [37].

3.2. Kernel matching

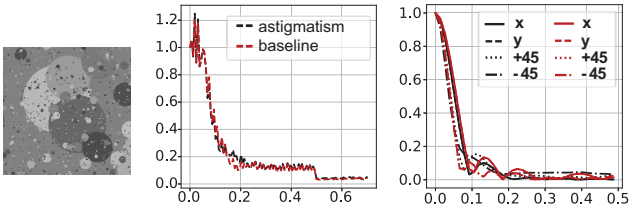
To compare the impact of different kernel types on each other, it is crucial to have size-matched representations. As a baseline, we use the simple disk-shaped kernel prototype from Hendrycks et al. [10] shown in Fig. 1g. Then, with an educated initial guess of coefficient values two kernels are evaluated on different metrics and optimized by offsetting the coefficients in steps of $\pm 0.1\lambda$. From this, we take the best overall fit as the kernel pair.

First, to compare two kernels, the Modulation Transfer Function (MTF) is obtained and evaluated in differently orientated slices (0°, 45°, 90°, 135°) [45, 39]. From this, the frequency value at 50% (MTF50) and the area under the curve (AUC) are obtained. The difference between these metrics provides a distance in optical quality. Further, SSIM and PSNR are reported [46]. These metrics aim to compare the shape. Secondly, the kernels are convolved and then analyzed on established test images used in benchmarking camera image quality as another matching criterion. [47, 48, 49] We evaluate SSIM and PSNR on a slanted edge test chart and the scale-invariant spilled coins test chart, both at the required ImageNet target resolution of 224×224. The spilled coins chart, shown in Fig. 4a, consists of randomly generated disks of different sizes and is designed to measure texture loss as an image-level MTF. [50, 51] We report here the acutance and MTF50 value as evaluated with the algorithm of Burns [51] from the spilled coins chart and compare once again between the two kernel representations. The evaluated image and optical quality on the spilled coins test chart serves here as proxy for other images from the image datasets.

Although the obtained optical kernels match also for higher severities with the defocus blur corruption [10], blur kernel sizes observed in the real world would be smaller, so this poses rather an upper bound on the observable blur corruption for an entire image. However, small objects such as pedestrians in object detection datasets such as [2] may suffer from such severe blurring as the object size decreases to tens of pixels.

4. OpticsBench - benchmarking robustness to selected optical aberrations

Similar to Hendrycks et al. [10] we define five different severities for each optical aberration. To obtain comparable corruptions, the defocus blur corruption from [10] serves as baseline for OpticsBench. For each severity, our kernels are matched to the defocus blur kernel types using the method and metrics described above. We generate two sets of kernels with different chromatic aberrations: OpticsBenchRG consists of the same kernels as OpticsBench, but with only the red and green channels spread as in Fig. 1b and thus producing severe chromatic aberration. For brevity, we refer



(a) (b) (c)

Figure 4: (a) Spilled coins test chart generated with Imatest target generator tool [52]. (b) MTF curves obtained from degraded versions of (a) and the corresponding PSF based MTF (c). The red curves refer to the baseline kernel at severity 3 and black to the corresponding astigmatism.

Table 2: Corruption types used in OpticsBench with their corresponding Zernike mode equivalent from Tab. 1.

Astigmatism	$(5, 6)$	Trefoil	$(10, 11)$
Coma	$(7, 8)$	Defocus & spherical	$(4, 9)$

to our kernels as *optical kernels*, although the disk-shaped kernel type from [10] is a model for defocus obtained from geometric optics.

Each set of kernels consists of 40 kernels for the eight different Zernike modes. We combine Zernike modes that result in similar shapes into a single corruption as in Tab. 2. This results in four different optical corruptions. Each corrupted dataset is then obtained by randomly assigning one of the two kernel types. A predefined seed of the pseudo-random number generator ensures reproducibility. Before applying the blur kernels, the images are resized to 256×256 and center cropped to 224×224, to avoid any reduction of the effect.

A set of DNNs is inferred for each severity and corruption, and the classification accuracy (acc@1) is reported. Additionally, the baseline and the clean dataset are evaluated.

To compare accuracies with the baseline, we define the deviation of a specific corruption c_i from the baseline b (defocus blur):

$$\Delta_{c_i,b} = Acc_{c_i} - Acc_b \tag{3}$$

Additionally, the Kendall Tau rank coefficient [53] is evaluated on the DNN model ranking to further investigate whether the optical kernels elicit different behavior compared to the disk-shaped kernels for different architectures.

The Python code to re-create both the benchmark inference scripts and the benchmark datasets for all optical corruptions is provided[1]. This includes the sets consisting of 40 kernels for OpticsBench and OpticsBenchRG. The script

[1] https://github.com/PatMue/classification_robustness

is intended for ImageNet-1k and ImageNet-100, but can be extended to other datasets. Since the benchmark is intended to be modifiable to specific user investigations (e.g. different aberrations), the source code to create kernels is also available but not as part of this contribution. Generating the OpticsBench datasets (five severities, four corruptions) from the 50k ImageNet-1k validation images takes about 120 minutes with six PyTorch workers and batch size 128 on an 8-core i7-CPU and 32GB RAM equipped with NVIDIA GeForce 3080-Ti 12GB GPU. Using smaller kernels than ours ($25 \times 25 \times 3$) may speed up the process.

5. Experiments on ImageNet

OpticsBench is built on the ImageNet validation dataset, consisting of 50k images distributed in 1000 classes. All four corruptions (astigmatism, coma, defocus blur & spherical and trefoil) are divided into 5 different severities each. We evaluate the 65 DNNs from the torchvision model zoo including a wide range of architectures as listed in Tab. 3. All models are pre-trained on ImageNet-1k. MobileNet_v3 is trained using AutoAugment [21], EffcientNet training uses CutMix [54] and MixUp [55] and ConvNeXt and VisionTransformers use a combination of these. In addition to these networks, we evaluate ResNet50 models from the RobustBench [27] leaderboard, which are reportedly robust against common corruptions [22, 23, 56]. Accuracies on the validation set are available in the supplementary material.

Table 3: Selected architectures used from PyTorch vision model zoo in small and large variants.

ConvNeXt [57]	Inception_v3 [58]	ResNet [59]
DenseNet [60]	MobileNet [61]	ResNeXt [62]
EfficientNet [63]	RegNet [64]	ViTransformer [65]

Fig. 5 shows the difference in accuracy compared to the validation data for all corruptions. The severities are plotted from left to right for a single corruption. The different colors refer to representatives from Table 3. The evaluation of all 70 models is found in the supplementary material A. In Fig. 5 the ResNet50 architecture, marked with red triangles for the standard training and with red circles for ResNet50+DeepAugment, clearly benefits from the training with DeepAugment. It drops by almost 65% points for severity 5 and astigmatism compared to the clean data accuracy. The robust model with DeepAugment looses 52% points. In general, the performance at a particular severity and corruption depends on the DNN.

Fig. 6 compares the achieved accuracy at a specific severity and mode with the disk-shaped kernel equivalent from [10]. For each severity the four OpticsBench corruptions are shown from left to right (astigmatism, coma, defocus & spherical, trefoil). The red circles show for the robust

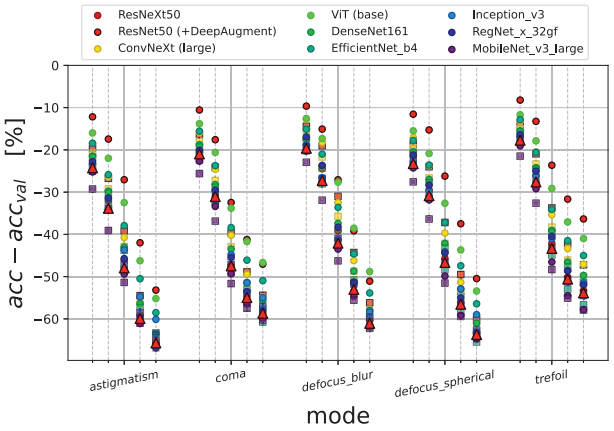


Figure 5: Difference in Accuracy on OpticsBench compared to the clean data. For each corruption five levels of severity are shown from left to right. We also include defocus blur [10], used as baseline to match our kernel sizes. Each color represents a DNN from Table 3.

ResNet50 model that the particular relative performance depends not only on the severity but also on the corruption: the various kernel types challenge each DNN differently. Especially the robust ResNet50 (red circle) and Inception (light blue circle) differ from the mean deviation. This is further justified by the ranking comparisons in the supplementary material A. The different rankings show that the blur types are processed differently by the DNNs and that each blur type needs to be considered.

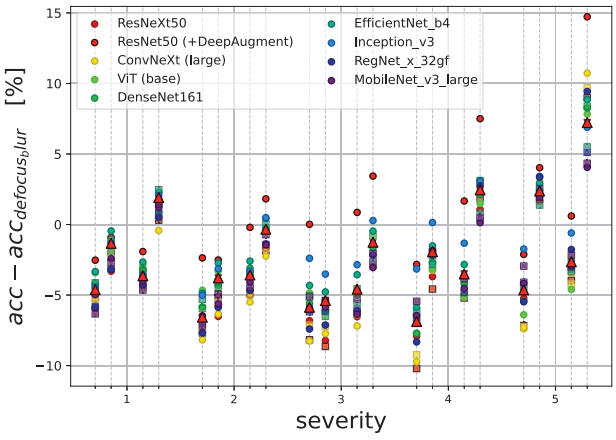


Figure 6: Comparison of Accuracy for the baseline and the different corruptions and severities. The corruptions per severity are from left to right: astigmatism, coma, defocus & spherical and trefoil.

6. OpticsAugment - augmenting with optical kernels

Since the above results show a clear drop in performance due to the different corruptions, and motivated by the general benefit of training with data augmentation, we propose here a data augmentation method with kernels from the OpticsBench kernel generation, which supplements existing methods.

OpticsAugment follows a similar approach as AugMix [22] and is described in Alg. 1. During dataloading in the training process each image is convolved with an individual RGB-kernel from the kernel stack containing e.g. 40 kernels for the different corruptions and severities. The random selection of the kernel follows a uniform distribution. Additionally, each resulting image is a weighted combination of the original image and the blurred image, following a beta-distribution as in [22], which controls the amount of augmentation. Thus, the total number of images per epoch stays the same. The exact combination varies each epoch, which facilitates generalization to the blur types. Since the processing is accelerated by GPU parallelization, the blurred images do not need to be computed in advance, but can be computed on-the-fly with comparable overhead to other methods such as [11] and a kernel size of 25x25x3. For this, Alg. 1 is implemented using parallelization on full image batches, which results in strong acceleration. For an image batch of 128 images an overhead of about 250ms is generated. Using dataloaders takes much longer since the number of workers is typically about 4-8, while the batch size is typically between 32 and 128. The implementation relies heavily on GPU acceleration using cuda and pytorch. The processed image batch is then normalized to the dataset specific mean and standard deviation. This is crucial, otherwise the training loss won't converge.

Since AugMix [22] provides low-cost data augmentation and promising results but no diverse blur kernel augmentation, we also try pipelining AugMix and OpticsAugment. Since both methods can feed the training algorithm with highly corrupted images, direct chaining leads to non-convergence and low accuracy. Therefore, we model the probability of augmentation with a flat Dirichlet distribution in four dimensions: The first two variables are the probabilities for AugMix and OpticsAugment. The other variables are auxiliary variables and ensure that the overall probability of each augmentation remains uniformly distributed. The output of OpticsAugment is normalized.

Algorithm 1: OpticsAugment

augment $(\boldsymbol{x}, kernels, severity = 3, \alpha = 1.0)$:
  Input : Clean tensor $\boldsymbol{x}$ consisting of N images, kernel stack $kernels$ and severity between 1-5, intensity α
  Output: Randomly blurred image batch $\boldsymbol{x}$
for $n = 0, 1, ..., N$ **do**
  $blurred = zeros\_like(x)$
  $h \leftarrow choice(kernels)$ randomly select a kernel from $kernels$ from one of the available augmentation types (e.g. oblique astigmatism).
  for *color=0,1,..,3* **do**
    $blurred[n, color] \leftarrow \boldsymbol{x}[n, color] * h[color]$ 2D convolution for image x and $color$
  end
  $p \leftarrow realization(\alpha)$ sample from a β-distribution controlling amount of augmentation
  $x \leftarrow (1 - p) \cdot x + p \cdot blurred$
end
$\boldsymbol{x} \leftarrow normalize(\boldsymbol{x})$ adapt image batch to dataset specific mean and standard deviation
return $\boldsymbol{x}$

7. Experiments on ImageNet-100

On ImageNet-100, which is a smaller dataset with only 100 classes from ImageNet-1k, five different DNNs are trained with OpticsAugment. In addition, we train baseline models on ImageNet-100 with the same hyperparameter settings, but without the data augmentation. These DNNs are then compared to each other on OpticsBench and 2D Common Corruptions [10] applied to ImageNet-100. We select five different architectures: EfficientNet_b0, MobileNet_v3_large, DenseNet161, ResNeXt50 and ResNet101.

The train split is divided into 5% validation images and 95% train images to ensure that the benchmark data only contains unseen data. On top of the trained DNNs all models are also trained with same settings but include OpticsAugment with severity 3 during training and the amount of augmentations is uniformly distributed, so $\alpha = 1.0$. The hyperparameter settings follow the standard training recipes as reported in the pytorch references [66] using cross-entropy loss, stochastic gradient descent and learning rate scheduling but no data augmentation. However, no additional data augmentation such as CutMix [54] or AutoAugment [21] are used. So, the achieved accuracy for MobileNet would be improved with AutoAugment. The DNNs are trained using the same batch size and number of epochs for clean and OpticsAugment training. Fine-tuning the hyperparameters can further improve the observed benefit.

Tab. 4 gives an overview of the improvement on ImageNet-100 OpticsBench with OpticsAugment. The smaller DNNs (MobileNet, EfficientNet_b0) show significantly lower improvements in accuracy, while ResNeXt50

Table 4: Performance *gain* with OpticsAugment on all ImageNet-100 OpticsBench corruptions. Average difference in accuracy across all corruptions in %-points for each severity. Details are given in supplementary B, tables 9-13.

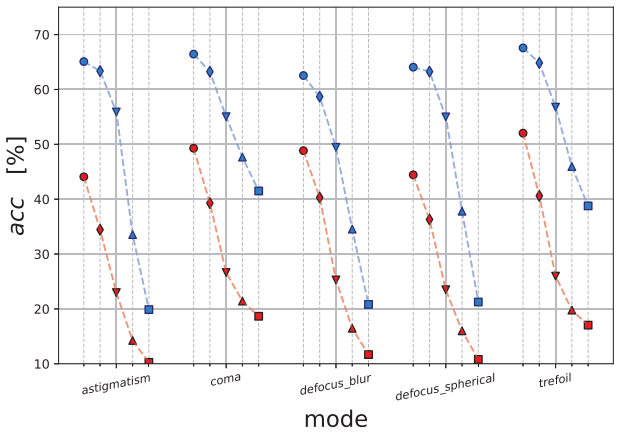
DNN	1	2	3	4	5
DenseNet161	14.77	21.96	27.26	20.98	13.84
ResNeXt50	17.40	24.49	29.56	22.32	14.76
ResNet101	9.97	16.24	21.15	17.39	12.07
MobileNet	8.12	12.73	13.80	10.09	7.27
EfficientNet	8.45	12.60	12.90	9.43	7.35

gains up to 29.6% points with OpticsAugment. Fig. 7 compares the accuracies with/without OpticsAugment during training for each corruption. The ResNeXt50 in Fig. 7a improves with OpticsAugment (blue) on average by 21.7% points compared to the default model (red). The robustness to coma is significantly improved, especially for higher severities. The results for DenseNet161 in Fig. 7b are similar with an average improvement of 19.8% points. The severity 3 accuracies in Fig. 7b for the default DenseNet161 model are comparable, while OpticsAugment is significantly more robust to OpticsBench corruptions. For instance, trefoil (last data point) is handled overly well while the performance gain for defocus blur is significantly lower. This seems to hold across different severities and DNNs: The corresponding disk-shaped kernel [10] was not present during training, suggesting that diverse blur kernels need to be considered during training. Together with tables supporting this claim the results for the other DNNs can be found in the supplementary material B.

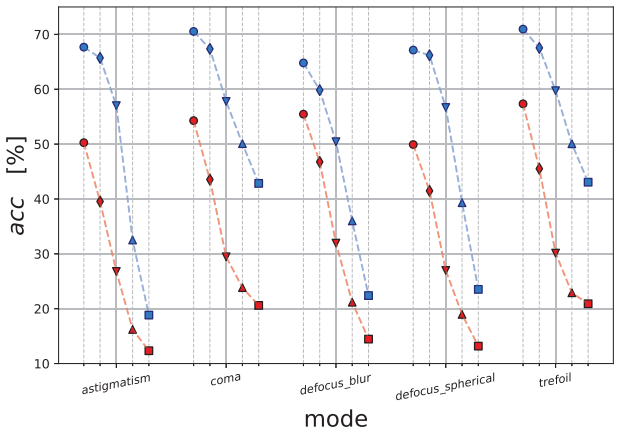
Additionally, the tuples of DNNs (baseline, baseline+OpticsAugment) are evaluated on 2D common corruptions [10]. The benchmark consists of 19 different corruptions including various blur types, noise and weather conditions. The average improvement with OpticsAugment is shown in Tab. 5. On average all DNNs still improve with OpticsAugment under the influence of the 2D common corruptions. As an example, we discuss here the results for ResNeXt50 with and without OpticsAugment in

Table 5: Performance *gain* with OpticsAugment on all 2D common corruptions [10] as average difference in accuracy across all corruptions in %-points for each severity. Details are given in supplementary B, tables 15-19.

DNN	1	2	3	4	5
DenseNet161	5.08	7.55	8.73	7.30	5.38
ResNeXt50	5.11	7.63	8.68	7.18	5.27
ResNet101	1.25	3.07	4.55	4.90	4.10
MobileNet	3.58	4.92	4.78	3.69	3.07
EfficientNet	4.35	6.32	6.70	4.62	3.69



(a) ResNext50



(b) DenseNet161

Figure 7: Accuracy on OpticsBench-ImageNet-100 for DNNs with (blue) and without (red) OpticsAugment training. The x-axis shows for each of the five corruptions (Astigmatism, Coma, Defocus [10], Defocus & Spherical, Trefoil) five different severities (from left to right). (a) ResNeXt50 (b) DenseNet161. The accuracy increases by on average 29.6% points for ResNeXt50 compared to the clean trained DNN (red) **with OpticsAugment (blue)** and severity 3 (down-pointing triangles). The performance gain for defocus blur [10] is the lowest for all severities.

Fig. 8. (Results for more models are in the supplementary material B). Fig. 8 compares the accuracies for ResNeXt50 solely trained on ImageNet-100 (red) and augmented with OpticsAugment (blue) respectively. For most of the corruptions the augmentation is beneficial, including blur, weather corruptions (fog, frost) and pixelation. However, some corruptions are not compensated by OpticsAugment, e.g. JPEG compression and contrast. Noise robustness may be improved by chaining OpticsAugment with AugMix.

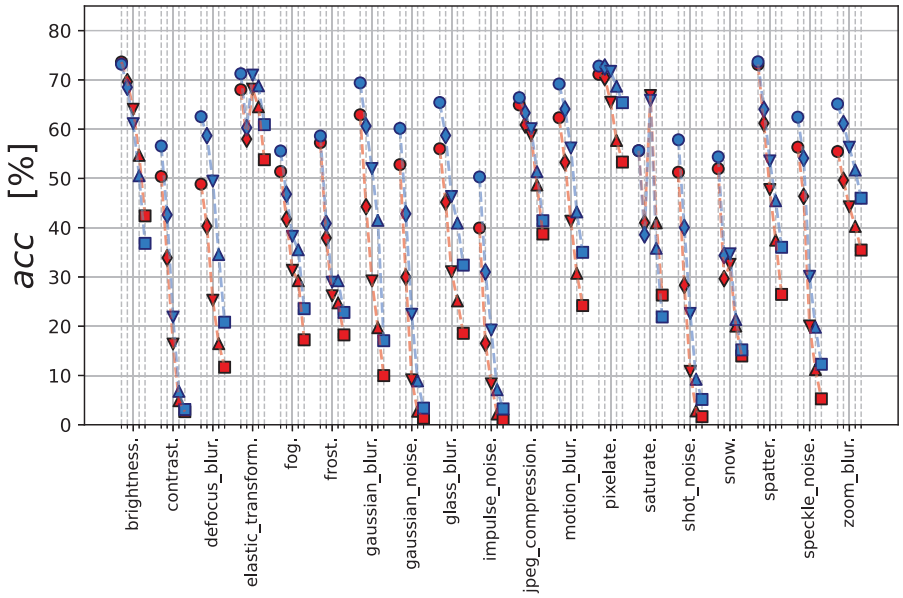


Figure 8: Accuracy for ResNeXt50 evaluated on ImageNet-100-C 2D common corruptions w/wo OpticsAugment training and all severities 1-5 (circle, diamond, triangles and square markers) at each corruption. **OpticsAugment (blue)** accuracy compared to the conventionally trained DNN (red). Results for more DNNs are available in supplementary B, Figs. 13- 15.

In addition, pipelining AugMix with OpticsAugment further improves robustness on 2D Common corruptions (especially to gaussian and speckle noise, cf. Table 25), but reduces the accuracy on OpticsBench. Table 6 shows the average improvement for a cascaded application of AugMix & OpticsAugment and the different severities. The accuracy of EfficientNet increases by an additional 3.4% points on average for 2D common corruptions.

Table 6: Additional average improvement for DNNs with pipelining evaluated on ImageNet-100-c 2D common corruptions [10].

DNN	1	2	3	4	5
EfficientNet	+2.79	+3.89	+4.34	+3.43	+2.61
MobileNet	+1.59	+2.27	+1.75	+0.57	-0.32

Limitations. First, with OpticsBench and the according augmentation, we make only one further step towards realistic benchmarking for robustness. As we can only test for sample classes of aberrations here, we point users to the kernel generation tool to test their specific use case. While we show that the proposed augmentation is beneficial for other types of common corruptions [10], we also evaluate towards adversarial robustness on ImageNet-100 test batches for l_2 bounded attacks from AutoAttack [15] and $\epsilon = 4/255$ in the supplementary material F, table 22. The evaluation is done for both APGD-CE and APGD-DLR attacks using the joint version of APGD and EoT for random defenses [67] and 5 restarts. However, the analysis does not indicate any benefits from OpticsAugment for adversarial robustness.

8. Conclusion

This paper proposes the use of optical kernels obtained from optics to benchmark model robustness to aberrations. These 3D kernels (x, y, color) have diverse shape compared to disk-shaped kernels and depend on color. To compare their influence to a baseline, they are matched to disk-shaped kernels by minimizing various optical and image quality metrics, and provide OpticsBench, a benchmark aimed at testing for lens aberrations. We show empirically on ImageNet that a large number of DNNs can handle the optical corruptions differently well and conclude that these diverse blur types should be considered.

In addition, we investigate a training method to achieve robustness on OpticsBench: OpticsAugment efficiently generates an augmentation for each image and epoch by randomly selecting a blur kernel and convolving it with the image. Augmentation with OpticsAugment is beneficial beyond OpticsBench, for example for different types of 2D common corruptions. An average performance gain of 6.8% compared to a DNN without augmentation is achieved across a large number of corruptions, with peaks of up to 29% at medium severities.

Acknowledgements The computations were supported by the OMNI Cluster of the University of Siegen.

References

[1] Claudio Michaelis et al. “Benchmarking Robustness in Object Detection: Autonomous Driving when Winter is Coming”. In: *arXiv:1907.07484 [cs, stat]* (Mar. 31, 2020). arXiv: 1907.07484.

[2] F. Yu et al. “BDD100K: A Diverse Driving Dataset for Heterogeneous Multitask Learning”. In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). ISSN: 2575-7075. June 2020, pp. 2633–2642.

[3] Holger Caesar et al. “nuScenes: A Multimodal Dataset for Autonomous Driving”. In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). Seattle, WA, USA: IEEE, June 2020, pp. 11618–11628.

[4] Geoffrey Hinton et al. “Deep Neural Networks for Acoustic Modeling in Speech Recognition: The Shared Views of Four Research Groups”. In: *IEEE Signal Processing Magazine* 29.6 (Nov. 2012). Conference Name: IEEE Signal Processing Magazine, pp. 82–97.

[5] Junyi Chai et al. “Deep learning in computer vision: A critical review of emerging techniques and application scenarios”. In: *Machine Learning with Applications* 6 (Dec. 15, 2021), p. 100134.

[6] Alan L. Yuille and Chenxi Liu. “Deep Nets: What have They Ever Done for Vision?” In: *International Journal of Computer Vision* 129.3 (Mar. 1, 2021), pp. 781–802.

[7] Nicholas Carlini and David Wagner. “Adversarial Examples Are Not Easily Detected: Bypassing Ten Detection Methods”. In: *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*. CCS ’17: 2017 ACM SIGSAC Conference on Computer and Communications Security. Dallas Texas USA: ACM, Nov. 3, 2017, pp. 3–14.

[8] Nicolas Papernot et al. “The Limitations of Deep Learning in Adversarial Settings”. In: *2016 IEEE European Symposium on Security and Privacy (EuroS&P)*. 2016 IEEE European Symposium on Security and Privacy (EuroS&P). Mar. 2016, pp. 372–387.

[9] Francesco Croce et al. *RobustBench: a standardized adversarial robustness benchmark*. Oct. 31, 2021. arXiv: 2010.09670[cs,stat].

[10] Dan Hendrycks and Thomas Dietterich. “Benchmarking Neural Network Robustness to Common Corruptions and Perturbations”. In: *Proceedings of the International Conference on Learning Representations* (2019).

[11] Oğuzhan Fatih Kar et al. “3D Common Corruptions and Data Augmentation”. en. In: 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). 2022, pp. 18963–18974.

[12] Alexander Braun. “Automotive mass production of camera systems: Linking image quality to AI performance”. In: *tm - Technisches Messen* (June 15, 2022). Publisher: Oldenbourg Wissenschaftsverlag.

[13] Igor Vasiljevic, Ayan Chakrabarti, and Gregory Shakhnarovich. “Examining the impact of blur on recognition by convolutional networks”. In: *arXiv preprint arXiv:1611.05760* (2016).

[14] Yinpeng Dong et al. “Benchmarking Robustness of 3D Object Detection to Common Corruptions”. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2023, pp. 1022–1032.

[15] Francesco Croce and Matthias Hein. “Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks”. In: *Proceedings of the 37th International Conference on Machine Learning*. International Conference on Machine Learning. ISSN: 2640-3498. PMLR, Nov. 21, 2020, pp. 2206–2216.

[16] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. “DeepFool: A Simple and Accurate Method to Fool Deep Neural Networks”. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Las Vegas, NV, USA: IEEE, June 2016, pp. 2574–2582.

[17] Maksym Andriushchenko et al. “Square Attack: A Query-Efficient Black-Box Adversarial Attack via Random Search”. In: *Computer Vision – ECCV 2020*. Ed. by Andrea Vedaldi et al. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2020, pp. 484–501.

[18] Peter Lorenz et al. “Is RobustBench/AutoAttack a suitable Benchmark for Adversarial Robustness?” In: *The AAAI-22 Workshop on Adversarial Machine Learning and Beyond*. 2022.

[19] Sven Gowal et al. “Improving robustness using generated data”. In: *Advances in Neural Information Processing Systems* 34 (2021).

[20] Robert Geirhos et al. “ImageNet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness”. In: International Conference on Learning Representations. Sept. 27, 2018.

[21] Ekin D. Cubuk et al. “AutoAugment: Learning Augmentation Strategies From Data”. In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). Long Beach, CA, USA: IEEE, June 2019, pp. 113–123.

[22] Dan Hendrycks* et al. “AugMix: A Simple Data Processing Method to Improve Robustness and Uncertainty”. In: International Conference on Learning Representations. 2020.

[23] Dan Hendrycks et al. “The Many Faces of Robustness: A Critical Analysis of Out-of-Distribution Generalization”. In: *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*. 2021 IEEE/CVF International Conference on Computer Vision (ICCV). Montreal, QC, Canada: IEEE, Oct. 2021, pp. 8320–8329.

[24] Tonmoy Saikia, Cordelia Schmid, and Thomas Brox. “Improving robustness against common corruptions with frequency biased models”. In: *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*. 2021 IEEE/CVF International Conference on Computer Vision (ICCV). Montreal, QC, Canada: IEEE, Oct. 2021, pp. 10191–10200.

[25] Hadi Salman et al. “Do adversarially robust imagenet models transfer better?” In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 3533–3545.

[26] Maura Pintor et al. “ImageNet-Patch: A dataset for benchmarking machine learning robustness against adversarial patches”. In: *Pattern Recognition* 134 (2023), p. 109064.

[27] Francesco Croce et al. “RobustBench: a standardized adversarial robustness benchmark”. In: *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*. 2021.

[28] Max Born and Emil Wolf. *Principles of optics: electromagnetic theory of propagation, interference and diffraction of light*. 7th expanded ed. Cambridge ; New York: Cambridge University Press, 1999. 952 pp.

[29] Robert J. Noll. “Zernike polynomials and atmospheric turbulence*”. In: *JOSA* 66.3 (Mar. 1, 1976). Publisher: Optica Publishing Group, pp. 207–211.

[30] von F. Zernike. “Beugungstheorie des schneidenverfahrens und seiner verbesserten form, der phasenkontrastmethode”. In: *Physica* 1.7 (May 1, 1934), pp. 689–704.

[31] Larry N. Thibos. “Retinal image quality for virtual eyes generated by a statistical model of ocular wavefront aberrations”. In: *Ophthalmic and Physiological Optics* 29.3 (2009). _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1475-1313.2009.00662.x, pp. 288–291.

[32] D.R. Iskander, M.J. Collins, and B. Davis. “Optimal modeling of corneal surfaces with Zernike polynomials”. In: *IEEE Transactions on Biomedical Engineering* 48.1 (Jan. 2001). Conference Name: IEEE Transactions on Biomedical Engineering, pp. 87–95.

[33] *OpticStudio — Optical, Illumination & Laser System Design Software - Zemax*. URL: https://www.zemax.com/products/opticstudio (visited on 01/15/2023).

[34] Benjamin P. Cumming and Min Gu. “Direct determination of aberration functions in microscopy by an artificial neural network”. In: *Optics Express* 28.10 (May 11, 2020), p. 14511.

[35] M. N’Diaye et al. “Calibration of quasi-static aberrations in exoplanet direct-imaging instruments with a Zernike phase-mask sensor”. In: *Astronomy & Astrophysics* 555 (July 2013), A94.

[36] R. G. Lane and M. Tallon. “Wave-front reconstruction using a Shack–Hartmann sensor”. In: *Applied Optics* 31.32 (Nov. 10, 1992), p. 6902.

[37] Brandon Dube. “prysm: A Python optics module”. In: *Journal of Open Source Software* 4.37 (May 9, 2019), p. 1352.

[38] H. Kirshner, D. Sage, and M. Unser. “3D PSF Models for Fluorescence Microscopy in ImageJ”. In: *Proceedings of the Twelfth International Conference on Methods and Applications of Fluorescence Spectroscopy, Imaging and Probes (MAF’11)*. Strasbourg, French Republic, Sept. 2011, p. 154.

[39] Joseph W. Goodman. *Introduction to Fourier optics*. Fourth edition. New York: W.H. Freeman, Macmillan Learning, 2017. 546 pp.

[40] Michael Hirsch et al. “Efficient filter flow for space-variant multiframe blind deconvolution”. In: *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. 2010 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). San Francisco, CA, USA: IEEE, June 2010, pp. 607–614.

[41] Patrick Müller and Alexander Braun. “Simulating optical properties to access novel metrological parameter ranges and the impact of different model approximations”. In: *2022 IEEE International Workshop on Metrology for Automotive (MetroAutomotive)*. 2022 IEEE International Workshop on Metrology for Automotive (MetroAutomotive). July 2022, pp. 133–138.

[42] M. Řeřábek and P. Páta. “The space variant PSF for deconvolution of wide-field astronomical images”. In: *Adaptive Optics Systems*. Ed. by Norbert Hubin, Claire E. Max, and Peter L. Wizinowich. Vol. 7015. International Society for Optics and Photonics. SPIE, 2008, 70152G.

[43] James G. Nagy and Dianne P. O’Leary. “Fast iterative image restoration with a spatially varying PSF”. In: Optical Science, Engineering and Instrumentation ’97. Ed. by Franklin T. Luk. San Diego, CA, United States, Oct. 24, 1997, p. 388.

[44] Vasudevan Lakshminarayanan and Andre Fleck. “Zernike polynomials: a guide”. In: *Journal of Modern Optics* 58.7 (Apr. 10, 2011), pp. 545–561.

[45] Glenn D. Boreman. *Modulation Transfer Function in Optical and Electro-Optical Systems*. SPIE, July 1, 2001.

[46] Z. Wang et al. “Image Quality Assessment: From Error Visibility to Structural Similarity”. In: *IEEE Transactions on Image Processing* 13.4 (Apr. 2004), pp. 600–612.

[47] Jonathan B. Phillips and Henrik Eliasson. *Camera Image Quality Benchmarking*. Newark, UNITED KINGDOM: John Wiley & Sons, Incorporated, 2018.

[48] “IEEE Standard for Camera Phone Image Quality”. In: *IEEE Std 1858-2016 (Incorporating IEEE Std 1858-2016/Cor 1-2017)* (May 2017). Conference Name: IEEE Std 1858-2016 (Incorporating IEEE Std 1858-2016/Cor 1-2017), pp. 1–146.

[49] *ISO12233:2017, Photography — Electronic still picture imaging — Resolution and spatial frequency responses*. Standard. Volume: 2017. Geneva, CH: International Organization for Standardization, 2017.

[50] Jon McElvain et al. “Texture-based measurement of spatial frequency response using the dead leaves target: extensions, and application to real camera systems”. In: IS&T/SPIE Electronic Imaging. Ed. by Francisco Imai, Nitin Sampat, and Feng Xiao. San Jose, California, Jan. 17, 2010, p. 75370D.

[51] Peter D. Burns. “Refined measurement of digital image texture loss”. In: IS&T/SPIE Electronic Imaging. Ed. by Peter D. Burns and Sophie Triantaphillidou. Burlingame, California, USA, Feb. 4, 2013, 86530H.

[52] *Imatest Target Generator — Imatest*. URL: https://www.imatest.com/product/imatest-target-generator/ (visited on 03/01/2023).

[53] M. G. Kendall. “The Treatment of Ties in Ranking Problems”. In: *Biometrika* 33.3 (1945). Publisher: [Oxford University Press, Biometrika Trust], pp. 239–251.

[54] Sangdoo Yun et al. “CutMix: Regularization Strategy to Train Strong Classifiers With Localizable Features”. In: *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. 2019 IEEE/CVF International Conference on Computer Vision (ICCV). Seoul, Korea (South): IEEE, Oct. 2019, pp. 6022–6031.

[55] Hongyi Zhang et al. *mixup: Beyond Empirical Risk Minimization*. Apr. 27, 2018. arXiv: 1710.09412[cs,stat].

[56] N. Benjamin Erichson et al. *NoisyMix: Boosting Model Robustness to Common Corruptions*. May 22, 2022. arXiv: 2202.01263[cs,stat].

[57] Zhuang Liu et al. “A ConvNet for the 2020s”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2022).

[58] Christian Szegedy et al. “Going deeper with convolutions”. In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). ISSN: 1063-6919. June 2015, pp. 1–9.

[59] Kaiming He et al. “Deep Residual Learning for Image Recognition”. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). ISSN: 1063-6919. June 2016, pp. 770–778.

[60] Gao Huang et al. “Densely Connected Convolutional Networks”. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2017, pp. 4700–4708.

[61] Andrew G. Howard et al. *MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications*. Apr. 16, 2017. arXiv: 1704.04861[cs].

[62] Saining Xie et al. “Aggregated Residual Transformations for Deep Neural Networks”. In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). ISSN: 1063-6919. July 2017, pp. 5987–5995.

[63] Mingxing Tan and Quoc Le. “EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks”. In: *Proceedings of the 36th International Conference on Machine Learning*. International Conference on Machine Learning. ISSN: 2640-3498. PMLR, May 24, 2019, pp. 6105–6114.

[64] Ilija Radosavovic et al. “Designing Network Design Spaces”. In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). Seattle, WA, USA: IEEE, June 2020, pp. 10425–10433.

[65] Rene Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. “Vision Transformers for Dense Prediction”. In: *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*. 2021 IEEE/CVF International Conference on Computer Vision (ICCV). Montreal, QC, Canada: IEEE, Oct. 2021, pp. 12159–12168.

[66] *vision/references/classification at v0.11.0 · pytorch/vision*. URL: https://github.com/pytorch/vision/tree/v0.11.0/references/classification (visited on 03/01/2023).

[67] Anish Athalye, Nicholas Carlini, and David Wagner. “Obfuscated Gradients Give a False Sense of Security: Circumventing Defenses to Adversarial Examples”. In: *Proceedings of the 35th International Conference on Machine Learning*. International Conference on Machine Learning. ISSN: 2640-3498. PMLR, July 3, 2018, pp. 274–283.