# Training on Foveated Images Improves Robustness to Adversarial Attacks

**Anonymous Author(s)**
Affiliation
Address
email

# Appendix

## A Preventing Gradient Obfuscation

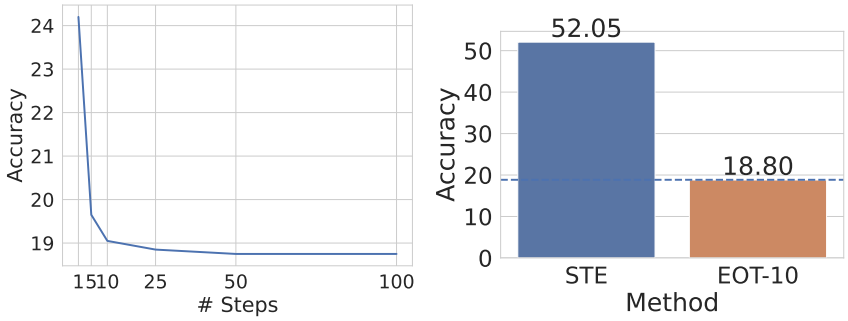
We take a number of measures to ensure that our results correspond to the true robustness of our method, and we avoid the pitfalls of gradient obfuscation [1, 2]. Firstly, we remove inference time stochasticity from all the models we test. We do this by sampling the Gaussian noise used in *R-Blur* and *VOneBlock* once and applying the same noise to all test images. Similarly, we sample the affine transform parameters for *RandAffine* once and use them for all test images. We also compute the fixation point sequences for *R-Blur* and *R-Warp* on unattacked images and do not update them during or after running APGD. Secondly, we ran APGD for 1 to 100 iterations and observed that as the number of iterations increases the success rate of the attack increases (Figure 1a). The success rate plateaus at 50 iterations. Since the attack success rate with 25 steps is only 0.1% lower than the success rate with 50 steps, we run APGD with 25 steps in most of our experiments. Thirdly, we applied expectation over transformation [3] by computing 10 gradient samples at each APGD iteration and averaging them to obtain the final update. We found this did not change the attack success rate so we take only 1 gradient sample in most of our experiments (Figure 1b). Finally, we also used a straight-though-estimator to pass gradients through *R-Blur* in case it may be obfuscating them and found that doing so reduces the attack success rate, thus indicating that gradients that pass through *R-Blur* retain valuable information that can be used by the adversarial attack (Figure 1b).

## B Fixation Point Selection

In this study, we did not attempt to develop an optimal fixation point selection algorithm, and instead, we operate under the assumption that points at which humans tend to fixate are sufficiently informative to perform accurate object classification. Therefore, we used DeepGaze-III [4], which is a neural network model trained to model the human gaze. DeepGaze-III uses a deep CNN backbone to extract features from the image, and based on these features another DNN predicts a heatmap that indicates, for each spatial coordinate, the probability that a human will fixate on it. However, it is possible that this algorithm is sub-optimal, and with further study, a better one could be developed. Though developing such an algorithm is out of the scope of this paper, we conduct a preliminary study to determine if it is possible to select better fixation points than the ones predicted by DeepGaze-III.

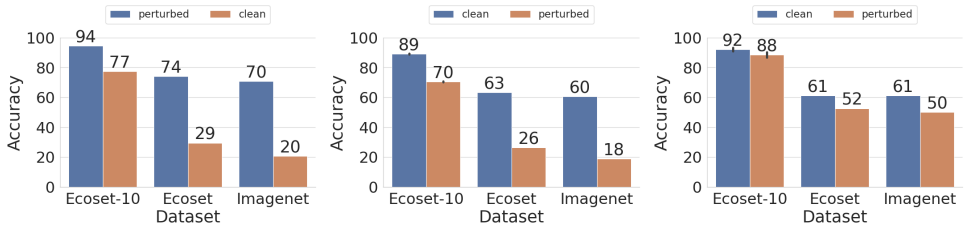
To this end, we run the following experiment to pick an optimal fixation point for each image during inference. For each testing image, we select 49 fixation points, spaced uniformly in a grid. Using the models we trained in earlier (see section 3) we obtain predictions for each image and each of the 49 fixation points. If there was at least one fixation point at which the model was able to correctly classify the image, we consider it to be correctly classified for the purpose of computing accuracy. We repeat this experiment for Ecoset-10, Ecoset, and Imagenet, using clean and adversarially perturbed data. We obtain the adversarially perturbed images for each of the 49 fixation points by fixing the

Submitted to 37th Conference on Neural Information Processing Systems (NeurIPS 2023). Do not distribute.




(a) (b)

Figure 1: Accuracy of a *R-Blur* model trained on Imagenet under APGD attack with different settings. (a) shows the accuracy when APGD attack is applied with different numbers of update steps. (b) shows the accuracy when 10 step of expectation-over-transformation (EOT-10) [3] is used and *R-Blur* is converted into a straight-through-estimator (STE) in the backward pass. The dashed line in (b) shows the accuracy of a 25-step APGD attack without EOT and normal gradient computation for *R-Blur*. Together these results strongly indicate that *R-Blur* does not obfuscate gradients and legitimately improves the adversarial robustness of the model.




(a) Optimal Fixation (b) DeepGaze-III Five Fixations (c) Adversarial Training

Figure 2: The accuracy obtained on clean and adversarial data when (a) the optimal fixation point was selected, (b) when the five fixation approach from Section 3 was used, and (c) an adversarially trained model was used.

fixation point at one location running the APGD attack with $\ell_\infty$-norm bounded to 0.004. Figure 3 illustrates this experiment with some example images.

The results are presented in Figure 2. We see that when the optimal fixation point is chosen accuracy on both clean and adversarially perturbed data improves, with the improvement in clean accuracy being the most marked. The clean accuracy on Ecoset-10, Ecoset, and Imagenet improved by 5%, 11%, and 10% respectively, which makes the clean accuracy of the *R-Blur* model on par or better than the clean accuracy achieved by the unmodified ResNet. Furthermore, when the optimal fixation point, is chosen *R-Blur* obtains higher clean accuracy than *AT* on all the datasets.

These results are meant to lay the groundwork for future work toward developing methods for determining the optimal fixation point based on the input image. However, they also illustrate that models trained with *R-Blur* learn features that are not only more adversarially robust features than ResNet but also allow the model to make highly accurate predictions on clean data.

## C Evaluations With Different Architectures

To demonstrate that the benefits of *R-Blur* are not limited to CNNs, we trained MLP-Mixer [5] and ViT [6] models with *R-Blur* preprocessing and evaluated their robustness. We use the configuration of MLP-Mixer referred to as S16 in [5]. Our ViT has a similar configuration, with 8 layers each having a hidden size of 512, an intermediate size of 2048, and 8 self-attention heads. We train both models with a batch size of 128 for 60 epochs on Ecoset-10 using the Adam optimizer. The learning

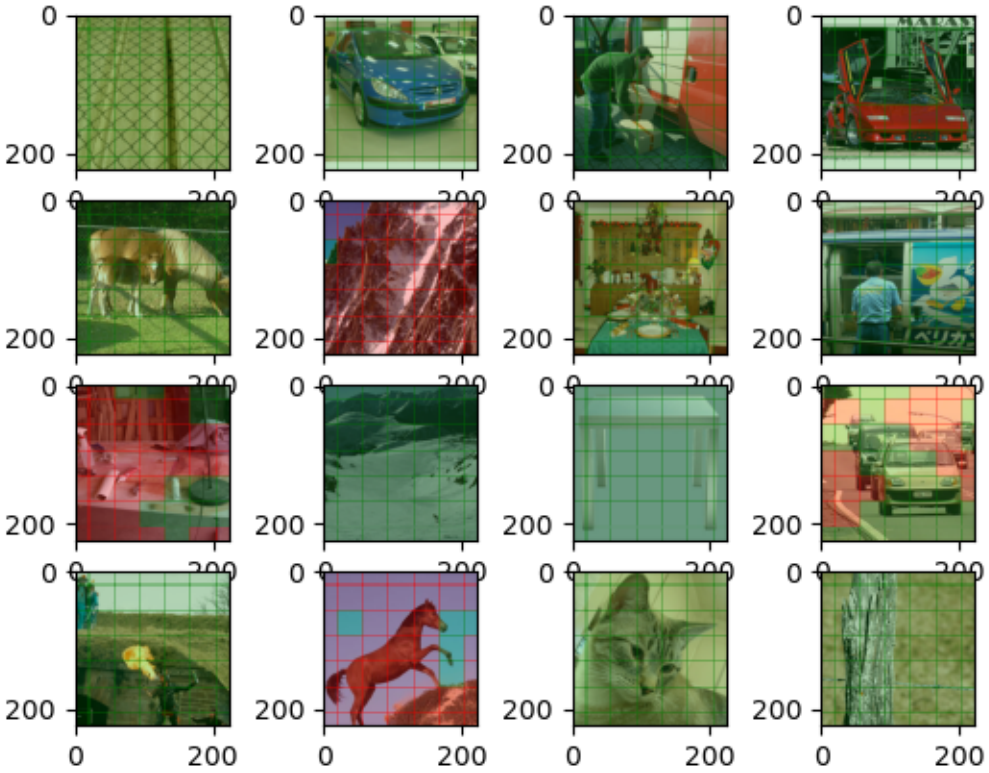


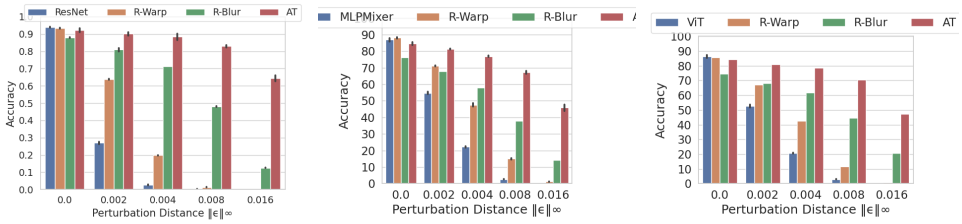

Figure 3: This figure indicates the locations of the optimal fixation points for some sample images. Each square in the grid corresponds to one of 49 fixation locations and represents the highest resolution region of the image if the model fixates at the center of the square. Squares that are shaded green indicate that the model's prediction at the corresponding fixation point was correct, while squares shaded red indicate that the model's prediction at the corresponding fixation point was incorrect. We see that there are certain images in which there are only a few optimal fixation points and they may not be in the center or in the corners of the image.

rate of the optimizer is linearly increased to 0.001 over 12 epochs and is decayed linearly to almost zero over the remaining epochs. The results are shown in Figure 4.

We observe that *R-Blur* significantly improves the robustness of MLP-Mixer models, and achieves greater accuracy than *R-Warp* at higher levels of perturbations. These results show that the robustness endowed to ResNets by *R-Blur* was not dependent on the model architecture, and they further strengthen our claim that loss in fidelity due to foveation contributes to the robustness of human and computer vision.

## D Breakdown of Accuracy Against Common Corruption by Corruption Type

In Figure 5 we break down the performance of the models on common corruptions by higher-level corruption categories. The individual members of each category are listed in Table 1. We see that in most of the categories, *R-Blur* achieves the highest median accuracy against the most severe corruptions. We also note that *R-Blur* exhibits a remarkable degree of robustness to noise, which is substantially greater than all the other models we evaluated. It is pertinent to note here that Gaussian noise was just 1 of the 4 types of noise included in the noise category, and thus the performance of *R-Blur* can not be attributed to overfitting on Gaussian noise during training. Furthermore, robustness to




(a) ResNet-18 (b) MLP-Mixer (c) ViT

Figure 4: The accuracy obtained on Ecoset-10 against adversarial perturbations of various $\ell_\infty$ norms when *R-Blur* is used with ResNet, MLP-Mixer and ViT backbones.

| Noise | Blur | Weather | Digital |
|---|---|---|---|
| gaussian noise | defocus blur | snow | contrast |
| shot noise | glass blur | frost | elastic transform |
| impulse noise | motion blur | fog | pixelate |
| speckle noise | zoom blur | brightness | jpeg compression |
| | gaussian blur | spatter | saturate |

Table 1: Categories of corruptions used to evaluate robustness to common corruptions. This categorization follows the one from [8]

one type of random noise does not typically generalize to other types of random noise [7]. Therefore, the fact that *R-Blur* exhibits improved robustness to multiple types of noise indicates that it is not just training on Gaussian noise, but rather the synergy of all the components of *R-Blur* that is likely the source of its superior robustness.

## E Sensitivity Analysis of Hyperparameters in *R-Blur*

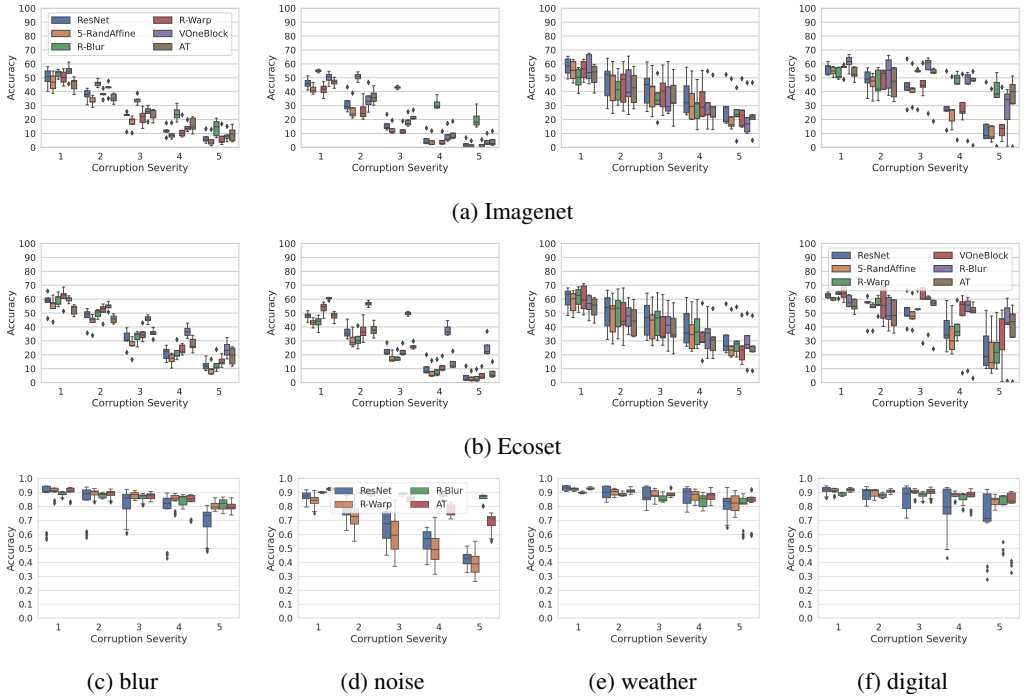
To measure the influence of the various Hyperparameters of *R-Blur* we conduct a sensitivity analysis. First, we vary the scale of the Gaussian noise added to the image, the viewing distance during inference, and the value of $\beta$ from Section 2.5, which is the scaling factor that maps eccentricity (see equation 1 to standard deviation, and measure the impact on accuracy on clean as well as adversarially perturbed data. The results of this analysis are presented in Figure 6. We see that, as expected, increasing the scale of the noise improves accuracy on adversarially perturbed data, however, this improvement does not significantly degrade clean accuracy. It appears that the adaptive blurring is mitigating the deleterious impact of Gaussian noise on clean accuracy. On the other hand, increasing $\beta$ beyond 0.01 surprisingly does not have a significant impact on accuracy and robustness. We also measured the accuracy on clean and perturbed data after varying the viewing distance (see 2.4) and the number of fixation points over which the logits are aggregated. These results are plotted in Figure 7, and they show that accuracy on clean and perturbed data is maximized when the width of the in-focus region is 48 (this corresponds to $vd = 3$) and aggregating over more fixation points improves accuracy on clean and perturbed data.

## F Training Configuration

Table 2 presents the configurations used to train the models used in our evaluation. For all the models the SGD optimizer was used with Nesterov momentum=0.9.

## G Implementation Details

We used Pytorch v1.11 and Python 3.9.12 to for our implementation. We used the implementation of Auto-PGD from the Torchattacks library (https://github.com/Harry24k/adversarial-attacks-pytorch). For *R-Warp* we used the code from the official repo https://github.com/mvuyyuru/adversary.git. Likewise, for *VOneBlock* we used the code from https://github.com/dicarlolab/vonenet, and




(a) Imagenet

(b) Ecoset

(c) blur (d) noise (e) weather (f) digital

(g) CIFAR-10

Figure 5: The accuracy achieved by *R-Blur* and baselines on various classes of common corruptions, proposed in [8]. The boxplot shows the distribution of accuracy values on 4-5 different corruptions in each class applied at different severity levels (x-axis) with 1 referring to least severe and 5 being the most severe corruption. *R-Blur* generally achieves the highest median accuracy on the highest severity levels.

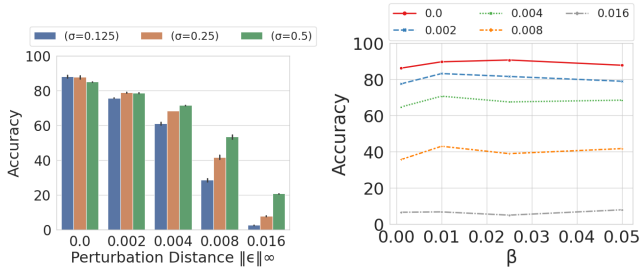



Figure 6: The impact of the hyperparameters of *R-Blur* on the accuracy and robustness of models trained on Ecoset-10. (left) the standard deviation of Gaussian noise, and (right) $\beta$ from Section 2.5.

for DeepGaze-III models we used the code from https://github.com/matthias-k/DeepGaze. The training code for DeepGaze-III with *R-Blur* and *R-Warp* backbones is based on https://github.com/matthias-k/DeepGaze/blob/main/train_deepgaze3.ipynb, and can be found in `adversarialML/biologically_inspired_models/src/fixation_prediction/train_deepgaze.py`. Our clones of these repositories are included in the supplementary material. For multi-gpu training, we used Pytorch Lightning v1.7.6. We used 16-bit mixed precision training to train most of our models. The code for *R-Blur* can be found in `adversarialML/biologically_inspired_models/src/retina_preproc.py` which is part of the supplemental material.

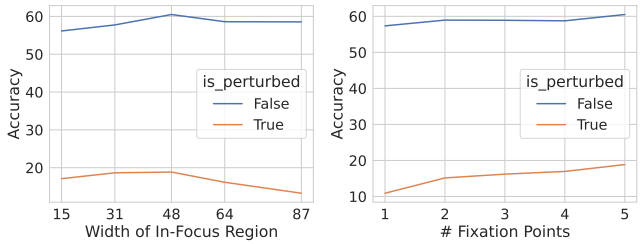



Figure 7: The impact of the size of the in-focus region by varying the viewing distance (left) and the number of fixation points over which the logits are aggregated (right) on accuracy. The plots are computed from a *R-Blur* model trained on Imagenet, and the perturbed data is obtained by conducting a 25-step APGD attack with $\|\delta\|_\infty = 0.004$. We see that accuracy on clean and perturbed data is maximized when the width of the in-focus region is 48 (this corresponds to $vd = 3$) and aggregating over more fixation points improves accuracy on clean and perturbed data.

| Dataset | Method | Batch Size | nEpochs | LR | LR-Schedule | Weight Decay | nGPUs |
|---|---|---|---|---|---|---|---|
| CIFAR-10 | ResNet | 128 | 0.4 | 60 | L-Warmup-Decay(0.2) | 5e-5 | 1 |
| | *AT* | 128 | 0.4 | 60 | L-Warmup-Decay(0.2) | 5e-5 | 1 |
| | *R-Warp* | 128 | 0.4 | 60 | L-Warmup-Decay(0.2) | 5e-5 | 1 |
| | *R-Blur* | 128 | 0.4 | 60 | L-Warmup-Decay(0.2) | 5e-5 | 1 |
| | *G-Noise* | 128 | 0.4 | 60 | L-Warmup-Decay(0.2) | 5e-5 | 1 |
| Ecoset-10 | ResNet | 128 | 0.4 | 60 | L-Warmup-Decay(0.2) | 5e-4 | 1 |
| | *AT* | 128 | 0.4 | 60 | L-Warmup-Decay(0.2) | 5e-4 | 1 |
| | *R-Warp* | 128 | 0.4 | 60 | L-Warmup-Decay(0.2) | 5e-4 | 1 |
| | *R-Blur* | 128 | 0.1 | 60 | L-Warmup-Decay(0.1) | 5e-4 | 1 |
| | *VOneBlock* | 128 | 0.1 | 60 | L-Warmup-Decay(0.1) | 5e-4 | 1 |
| | *G-Noise* | 128 | 0.4 | 60 | L-Warmup-Decay(0.2) | 5e-4 | 1 |
| Ecoset | ResNet | 256 | 0.2 | 25 | L-Warmup-Decay(0.2) | 5e-4 | 2 |
| | *AT* | 256 | 0.2 | 25 | L-Warmup-Decay(0.2) | 5e-4 | 4 |
| | *R-Warp* | 256 | 0.1 | 25 | L-Warmup-Decay(0.2) | 5e-4 | 4 |
| | *R-Blur* | 256 | 0.1 | 25 | C-Warmup-2xDecay(0.1) | 5e-4 | 4 |
| | *VOneBlock* | 256 | 0.1 | 25 | C-Warmup-2xDecay(0.1) | 5e-4 | 4 |
| | *G-Noise* | 256 | 0.1 | 25 | C-Warmup-2xDecay(0.1) | 5e-4 | 4 |
| Imagenet | ResNet | 256 | 0.2 | 25 | L-Warmup-Decay(0.2) | 5e-4 | 2 |
| | *AT* | 256 | 0.2 | 25 | L-Warmup-Decay(0.2) | 5e-4 | 4 |
| | *R-Warp* | 256 | 0.1 | 25 | L-Warmup-Decay(0.2) | 5e-4 | 4 |
| | *R-Blur* | 256 | 0.1 | 25 | C-Warmup-2xDecay(0.1) | 5e-4 | 4 |
| | *VOneBlock* | 256 | 0.1 | 25 | C-Warmup-2xDecay(0.1) | 5e-4 | 4 |
| | *G-Noise* | 256 | 0.1 | 25 | C-Warmup-2xDecay(0.1) | 5e-4 | 4 |

Table 2: The configurations used to train the models used in our evaluation. L-Warmup-Decay($f$) represents a schedule that linearly warms up and decays the learning rate and $f$ represents the fraction of iterations devoted to warmup. C-Warmup-2xDecay(0.1) is similar except that the warmup and decay follow a cosine function, and there are two decay phases. Both the schedulers are implemented using `torch.optim.lr_scheduler.OneCycleLR` from Pytorch.

## H Hardware Details

We trained our models on compute clusters with Nvidia GeForce 2080 Ti and V100 GPUs. Most of the Imagenet and Ecoset models were trained and evaluated on the V100s, while the CIFAR-10 and Ecoset-10 models were trained and evaluated on the 2080 Ti's.

## References


[1] Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *International conference on machine learning*, pages 274–283. PMLR, 2018.

[2] Nicholas Carlini, Anish Athalye, Nicolas Papernot, Wieland Brendel, Jonas Rauber, Dimitris Tsipras, Ian Goodfellow, Aleksander Madry, and Alexey Kurakin. On evaluating adversarial robustness. *arXiv preprint arXiv:1902.06705*, 2019.

[3] Anish Athalye, Logan Engstrom, Andrew Ilyas, and Kevin Kwok. Synthesizing robust adversarial examples. In *International conference on machine learning*, pages 284–293. PMLR, 2018.

[4] Matthias Kümmerer, Matthias Bethge, and Thomas SA Wallis. Deepgaze iii: Modeling free-viewing human scanpaths with deep learning. *Journal of Vision*, 22(5):7–7, 2022.

[5] Ilya O Tolstikhin, Neil Houlsby, Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Thomas Unterthiner, Jessica Yung, Andreas Steiner, Daniel Keysers, Jakob Uszkoreit, et al. Mlp-mixer: An all-mlp architecture for vision. *Advances in Neural Information Processing Systems*, 34:24261–24272, 2021.

[6] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[7] Robert Geirhos, Carlos RM Temme, Jonas Rauber, Heiko H Schütt, Matthias Bethge, and Felix A Wichmann. Generalisation in humans and deep neural networks. *Advances in neural information processing systems*, 31, 2018.

[8] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *arXiv preprint arXiv:1903.12261*, 2019.