A New Dataset Based on Images Taken by Blind People for Testing the Robustness of Image Classification Models Trained for ImageNet Categories - Supplementary Materials

Reza Akbarian Bafghi
University of Colorado, Boulder
reza.akbarianbafghi@colorado.edu

Danna Gurari
University of Colorado, Boulder
danna.gurari@colorado.edu

A. Appendix

This document supplements the main paper as follows:

1. Describes the reasons for excluding some categories and provides all selected categories of our dataset (supplements **Section 3.1.1**).
2. More details about the user interface that was used for the annotation task (supplements **Section 3.1.2**).
3. Explains our quality control protocols during the data collection process (supplements **Section 3.1.2**).
4. More details about the models we leveraged for benchmarking along with their accuracies on ImageNet, ImageNet-C, and VizWiz-Classification (supplements **Section 4**).
5. More details about the performance gap and the effective robustness (supplements **Section 4**).
6. Comparison of the performance of models evaluated on VizWiz-Classification and ObjectNet (supplements **Section 4**).

B. Dataset Creation

Removed Categories. As mentioned in the main paper, we intentionally excluded several ImageNet categories from our dataset. First, we did not include categories for which their meanings in ImageNet were distinct from their meanings when used in captions about the VizWiz images. For example, we removed "spotlight / spot" since in image captions "spot" mostly refers to the flash when capturing a picture. We also removed "nail / pin"" because "nail" often refers to a part of fingers in image captions. We also excluded from our dataset different species of cats and dogs since identifying those requires domain expertise; e.g., "Labrador retriever", "Golden retriever", "Cardigan Welsh corgi", "Boxer", "German shepherd dog", "Tabby, tabby cat", "Chihuahua", and "Egyptian cat". We also excluded categories which were not observed in at least 4 images, such as "church building" and "dock / dockage".

Selected Categories. Here, we list all categories included in our dataset: "Crock Pot", "desk", "desktop computer", "diaper / nappy", "digital clock", "digital watch", "dishwasher", "doormat", "electric fan", "electric guitar", "envelope", "espresso maker", "file / file cabinet", "fire screen / fireguard", "folding chair", "fountain", "frying pan / skillet", "goblet", "grand piano / grand", "grocery store", "hair spray", "hamper", "handkerchief / hankie", "hook / claw", "iPod", "iron / smoothing iron", "jeans", "t-shirt", "joystick", "lampshade / lamp shade", "laptop", "library", "lighter / igniter", "lipstick", "Loafer", "lotion", "speaker system", "mailbox", "measuring cup", "microphone", "microwave", "mixing bowl", "modem", "monitor", "computer mouse", "necklace", "packet", "pajama", "paper towel", "patio / terrace", "pedestal / footstall", "perfume / essence", "picket fence / paling", "pickup truck", "piggy bank", "pill bottle", "pillow", "pitcher", "plastic bag", "pole", "soda bottle", "pot / flowerpot", "printer", "projector", "purse", "quilt / comforter", "race car", "radiator", "radio / wireless (device)", "refrigerator / icebox", "remote control", "restaurant", "rocking chair / rocker", "rule / ruler", "running shoe", "saltshaker / salt shaker", "sandal", "scale / weighing machine", "screwdriver", "seat belt / seatbelt", "shopping cart", "shower curtain", "sliding door", "lion", "soap dispenser", "soccer ball", "sock", "tiger / Panthera tigris", "ice bear / polar bear", "soup bowl", "space heater", "sport car", "stage", "fly", "bee", "stove", "strainer", "studio couch", "suit / suit of clothes", "sunglasses", "sunscreen", "sweatshirt", "swing", "table lamp", "tape player", "teapot", "teddy / teddy bear", "television", "hog (animal)", "tile roof", "toaster", "toilet seat", "trailer truck / rig", "tray", "tub / vat", "umbrella", "upright / upright piano", "vacuum / vacuum cleaner", "vase", "vending machine", "wall clock", "wallet", "wardrobe /

closet / press", "washbasin", "washing machine", "water bottle", "water jug", "window screen", "window shade", "Windsor tie", "wine bottle", "wok", "wooden spoon", "acoustic guitar", "comic book", "street sign", "analog clock", "menu", "plate", "trash can", "book jacket / dust cover", "backpack", "ice cream", "ballpoint pen", "pretzel", "cheeseburger", "hotdog", "mashed potato", "broccoli", "cauliflower", "barrel", "baseball", "basketball", "cucumber", "artichoke", "bell pepper", "bath towel", "mushroom", "bathtub / bath", "strawberry", "orange (fruit)", "lemon", "beer bottle", "pineapple / ananas", "banana", "Granny Smith", "pomegranate", "binder / ring-binder", "dough", "pizza", "bookcase", "red wine", "espresso", "cup", "bow tie", "bucket", "buckle", "valley / vale", "baseball player", "candle / taper / wax light", "can opener / tin opener", "cardigan", "corn", "car mirror", "carton", "car wheel", "cassette", "cassette player", "CD player", "mobile phone", "toilet tissue", "chain", "chainlink fence", "chest / box", "chiffonier / commode", "bell / gong", "china closet", "Christmas stocking", "coffee mug", and "computer keyboard".

Their WordNet [5] IDs are as follow: "n07747607", "n03400231", "n04344873", "n03935335", "n04357314", "n04507155", "n04118776", "n03485794", "n04330267", "n02965783", "n03958227", "n04009552", "n02948072", "n02909870", "n04371774", "n04596742", "n04554684", "n04141975", "n04398044", "n03063599", "n07720875", "n03777754", "n15075141", "n04525305", "n07697537", "n04560804", "n04404412", "n02978881", "n04131690", "n03443371", "n03991062", "n07753275", "n02992529", "n07892512", "n04037443", "n04081281", "n04442312", "n07860988", "n03000134", "n07718747", "n03937543", "n04591713", "n03180011", "n03271574", "n03297495", "n03903868", "n02676566", "n03179701", "n03930313", "n09468604", "n03207941", "n04162706", "n03793489", "n12144580", "n02795169", "n04296562", "n03476991", "n04380533", "n02974003", "n03197337", "n03661043", "n03887697", "n04553703", "n03976657", "n02190166", "n04026417", "n02395406", "n03223299", "n03347037", "n03188531", "n02129604", "n03584829", "n03017168", "n04392985", "n03950228", "n03733805", "n07579787", "n03761084", "n03680355", "n02769748", "n04557648", "n07745940", "n02134084", "n04515003", "n04254680", "n04332243", "n04597913", "n02951585", "n04447861", "n03983396", "n02747177", "n07749582", "n04033995", "n03759954", "n02971356", "n07734744", "n04399382", "n03595614", "n07930864", "n07565083", "n04356056", "n03594734", "n04074963", "n03133878", "n02999410", "n04591157", "n07753592", "n07873807", "n03272010", "n03026506", "n03337140", "n04263257", "n03930630", "n03388043", "n06596364", "n03877472", "n03018349", "n03710193", "n03016953", "n03532672", "n04467665", "n02988304", "n04370456", "n03642806", "n07920052", "n04493381", "n03085013", "n02129165", "n04154565", "n04265275", "n03637318", "n04120489", "n03482405", "n03899768", "n04589890", "n06794110", "n03452741", "n02870880", "n04285008", "n02808440", "n04548280", "n04070727", "n04254120", "n04204347", "n03690938", "n02883205", "n09835506", "n03691459", "n03782006", "n07715103", "n03602883", "n03871628", "n04133789", "n04548362", "n04590129", "n02808304", "n03938244", "n04041544", "n07742313", "n04254777", "n07695742", "n03584254", "n07714990", "n07697313", "n03461385", "n03916031", "n07711569", "n07768694", "n04099969", "n04435653", "n03666591", "n03775546", "n04209239", "n04517823", "n04476259", "n04040759", "n04004767", "n02963159", "n04522168", "n03291819", "n02979186", "n03814906", "n02206856", "n03196217", "n03376595", "n04239074", "n07718472", "n02910353", "n02783161", "n04550184", "n03676483", "n02708093", "n03014705", "n07248320", "n07614500", "n02823428", "n02802426", "n04350905", "n02799071", and "n02840245".

User Interfaces for Annotation Task. We show a screenshot of our user interface for our annotation task, with instructions shown in Figure 1 and the annotation task in Figure 2. The task itself displays an image with 10 suggested object categories to select from. After selecting which categories are present from the predefined categories, the annotator must then indicate if any additional object categories are present.

We designed another interface for the authors to review images that workers determined have additional object categories beyond the ones the workers selected from. A screenshot of this interface is shown in Figure 3. Authors of the paper reviewed all relevant images with this interface. For each image, the authors first confirmed labels selected by AMT workers were indeed present, then indicated whether any of the other 218 relevant ImageNet categories were present, and finally identified the category of the main object of the image (when a single most prominent object was present in the image).

Quality Control. We performed quality control as we collected annotations from our workers to ensure continued high-quality results. The authors reviewed a subset of the results from crowdworkers who had submitted more than 10 HITs in each batch. The following mechanisms were used to determine which workers' answers to review:

- Workers who were a statistical outlier in time-to-submit. In this case, an outlier is a point that falls more than 1.5 times the interquartile range (IQR) above the third quartile or below the first quartile.
- Workers who, at least for one HIT, submitted answers, more than one-third (4 answers from 12 possible answers)

Instructions

Motivation:
Our goal is to design algorithms to assist people with vision impairments to learn what objects are in the pictures they take. We will use your work to help design such algorithms.

Task:
We ask that you help us identify what objects are present in the image.

To complete this task, please follow these two steps:

1. Review the image and then select all objects in the shown list that are observed in the image.
2. Indicate whether additional objects can be observed in the image beyond those already identified in Step 1.

Examples:
Regardless of occlusions, other objects, and clutter or text in the scene, select the appropriate label(s) if the image contains the object(s).

Qusestion 1:

- [ ] Desktop Computer
- [x] Laptop
- [ ] Computer Keyboard
- [x] Cup
- [ ] Monitor
- [ ] None of the Above

Qusestion 2:

- [x] Yes
- [ ] No

Fig. 1

If any of the objects in the Fig. 1 were not mentioned in Step 1, you should select "Yes" in Step 2. For the example above, we can observe there is also a "desk" in the image therefore the answer to the question 2 is "Yes".

Select objects that are found both its original physical form and in print.

Qusestion 1:

- [x] Pizza
- [ ] Cup
- [x] Flag
- [ ] Computer Mouse
- [x] Box
- [ ] None of the Above

Qusestion 2:

- [ ] Yes
- [x] No

Fig. 2

Figure 1. The instructions that we provided for AMT annotators.

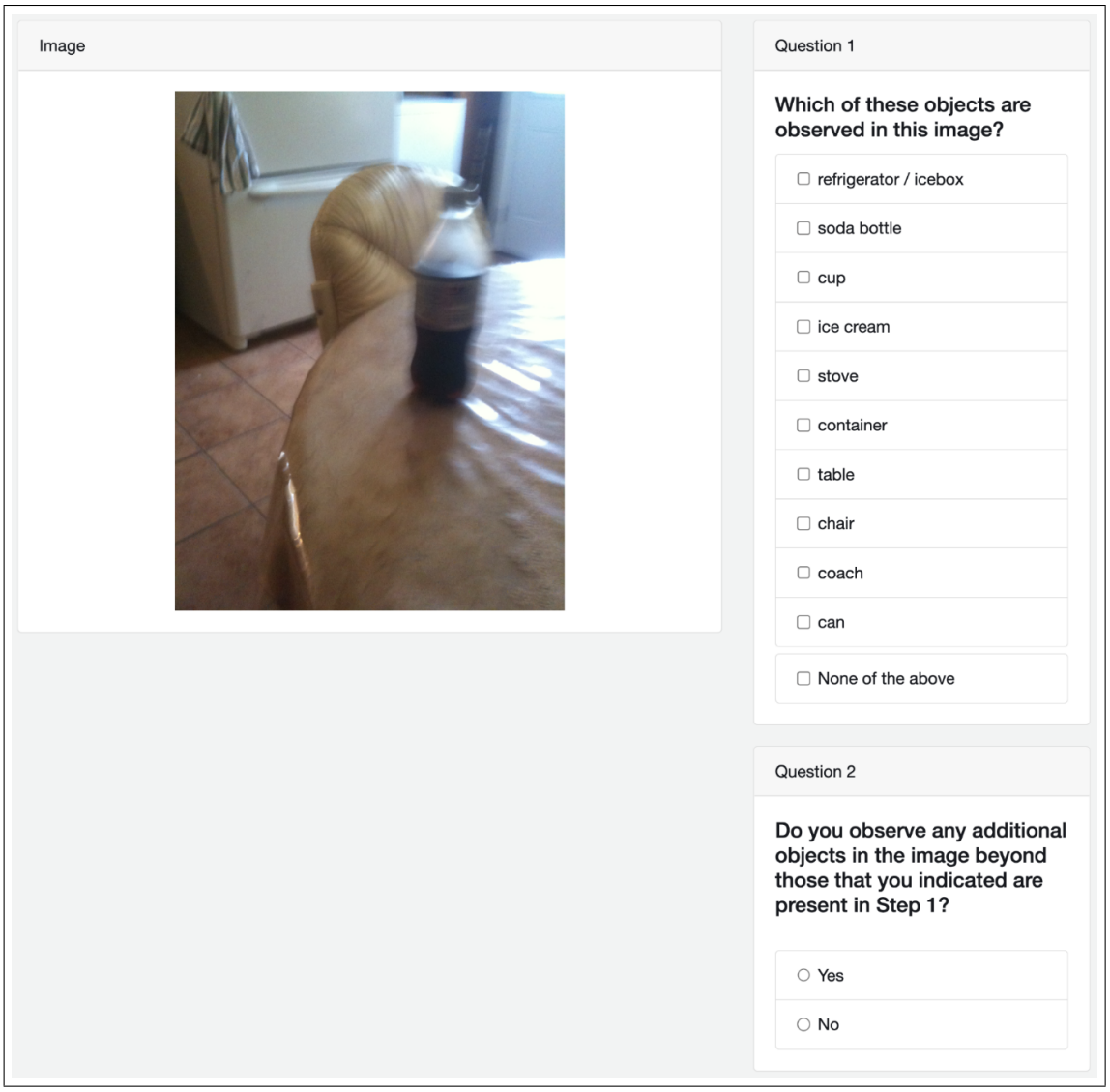

Figure 2. The user interface of our main annotation task.

of which were different from answers that the other two workers submitted.

- Workers who submitted answer "None of the above" to question 1 for more than 5% of HITs. (This is the percentage of HITs in the second phase of the qualification task, the answer to question 1 of which is "None of the above".)

- Workers who submitted answer "Yes" to question 2 for more than 30% of HITs. (This is the percentage of HITs in the second phase of the qualification task, the answer to question 2 of which is "Yes".)

C. Algorithm Benchmarking

For categorizing models and comparing accuracies, we used the test bed provided by [17] and PyTorch Image Models [18]. Table 1 shows the accuracy of 100 models tested on VizWiz-Classification as well as the accuracy of them on all images of ImageNet and ImageNet-C. In the following paragraphs, we provide more information about effective robustness and performance gap. Also, we explain models in detail.

Effective Robustness Inspired by the metric introduced in [17], the effective robustness of model x with respect to the accuracy of the models on dataset A and dataset B can

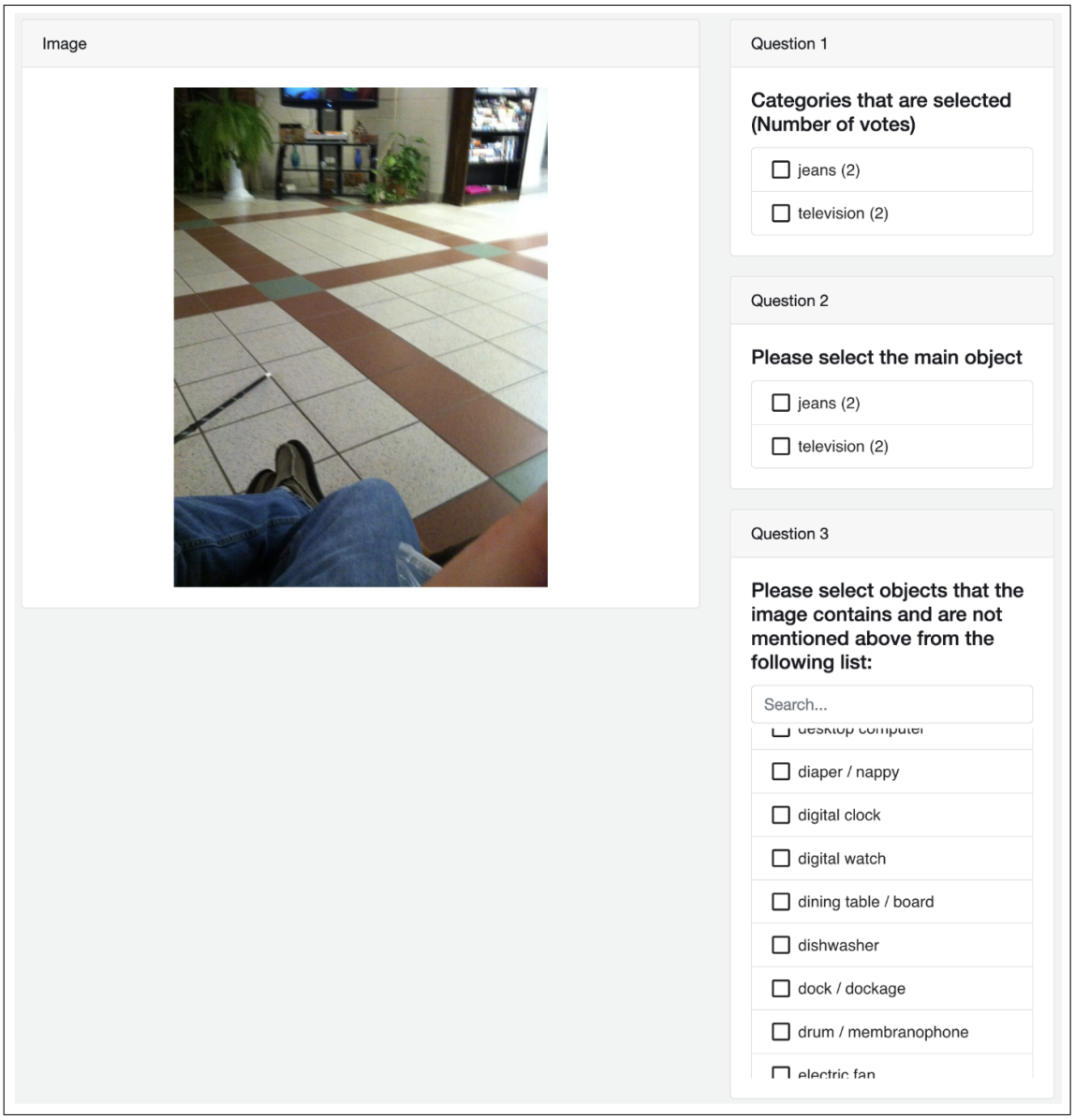

Figure 3. The user interface used by the authors to review results submitted by AMT workers.

be calculated as follows:

$$B(x) - (aA(x) + b)$$

where a and b are the slope and the y-intercept of the linear trend (accuracy of models on dataset A and B are on the x-axis and the y-axis respectively) and $A(x)$ and $B(x)$ are the accuracy of the model of x on datasets of A and B respectively. We split our models into three groups. Thus, to measure which group produces the most effective robustness, we can use this metric and take the average of it for each subgroup (i.e. models with robustness intervention), which can be calculated as:

$$\frac{1}{N}\sum_{i=1}^{N}[B(x_i) - (aA(x_i) + b)]$$

where $\{x_1, x_2, \ldots, x_N\}$ is the set of our models in a subclass.

Performance Gap. The performance gap of models from dataset A to dataset B can be calculated as below:

$$\frac{1}{N}\sum_{i=1}^{N}[A(x_i) - B(x_i)]$$

where $\{x_1, x_2, \ldots, x_N\}$ is the set of our models, and $A(x_i)$ and $B(x_i)$ are the accuracy of the model of x_i on datasets of A and B respectively. This equation averages the difference between the accuracy of the models on dataset B and the expected accuracy (accuracy of the models on dataset A).

Standard Models. Different settings of VGG [15], ResNet [7], and EfficientNet [16] are included in this group. Also, we test Vision Outlooker (VOLO) [21], which does not use any external data. EfficientNet models may leverage AutoAugment [1] and RandAugment [2]. Although they are data augmentation methods, because they are applied in standard versions of EfficientNet, we count them in this group.

Robust Models. Models of this group may use robustness intervention methods consisting of AdvProp [19], AugMix [10], DeepAugment [9], anti-aliasing [22], etc. Also, models that are trained on corrupted images (which are randomly sampled from different corruptions of ImageNet-C) [17], stylized images [6], and adversarial examples [4] of ImageNet images are included in this category.

Models Trained With More Data. ViT [3], ConvNeXt [11], and ResNeXt [20] are architectures included in this class. ConvNeXt and ViT models of our experiments are pre-trained on 14 million images of ImageNet-21k [13] with 21,843 classes and then fine-tuned on ImageNet. The ResNeXt model weakly-supervised pre-trained on 940 million images of the Instagram dataset with about 1,500 labels and fine-tuned on ImageNet images.

C.1. Comparing the Performance of Models Evaluated on VizWiz-Classification and ObjectNet

A comparison is shown in Figure 4. The majority of models are positioned above the $x = y$ line suggesting that ObjectNet is more challenging than our dataset. The range of accuracy drops is from -8.8% to 7.9%. Also, the performance gap is -3.4%. Although the robustness of models is similar in this comparison, models trained on more data performed slightly better. In general, ObjectNet is contrived with workers hired to take pictures of objects in pre-defined rotations, viewpoints, and backgrounds, while our dataset includes images from an authentic use case.

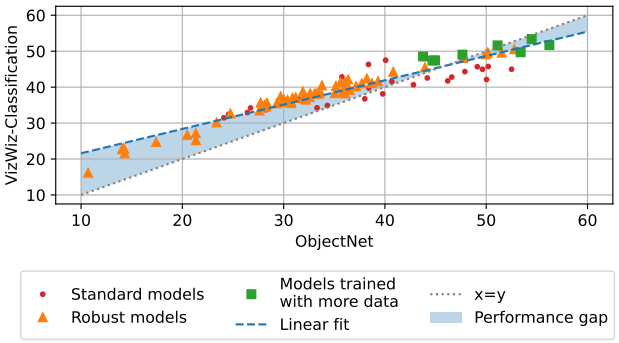

Figure 4. Comparing model accuracies on ObjectNet (x-axis) and VizWiz-Classification (y-axis).

References

[1] Ekin D. Cubuk, Barret Zoph, Dandelion Mané, Vijay Vasudevan, and Quoc V. Le. Autoaugment: Learning augmentation strategies from data. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 113–123. Computer Vision Foundation / IEEE, 2019. 6, 7, 8

[2] Ekin Dogus Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V. Le. Randaugment: Practical automated data augmentation with a reduced search space. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 3008–3017, 2020. 6, 7

[3] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *ArXiv*, abs/2010.11929, 2021. 6, 7

[4] Logan Engstrom, Andrew Ilyas, Hadi Salman, Shibani Santurkar, and Dimitris Tsipras. Robustness (python library), 2019. 6, 8

[5] Christiane Fellbaum. *WordNet: An Electronic Lexical Database*. Bradford Books, 1998. 2

[6] 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 *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. 6, 7, 8

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 770–778. IEEE Computer Society, 2016. 6, 7, 8

[8] Tong He, Zhi Zhang, Hang Zhang, Zhongyue Zhang, Junyuan Xie, and Mu Li. Bag of tricks for image classification with convolutional neural networks. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 558–567, 2019. 7

[9] Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Lixuan Zhu, Samyak Parajuli, Mike Guo, Dawn Xiaodong Song, Jacob Steinhardt, and Justin Gilmer. The many faces of robustness: A critical analysis of out-of-distribution generalization. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 8320–8329, 2021. 6, 7

Network	Type	IN	IN-C	VizWiz-Classification		
				Corr.	Clean	All
volo-d5-512 [21]	Standard	87.0	-	51.8	59.7	57.2
convnext-base-in22ft1k [11]	More data	85.8	-	46.9	56.0	53.5
vit-large-patch16-384 [3]	More data	87.1	-	46.8	56.2	53.4
ig-resnext101-32x16d [12, 20]	More data	84.2	60.5	48.1	54.8	51.7
vit-base-patch16-384 [3]	More data	86.1	-	44.6	54.9	51.6
efficientnet-b8-advprop-autoaug [19]	Robust	85.4	66.3	45.8	53.2	50.5
ig-resnext101-32x8d [12, 20]	More data	82.7	58.1	44.9	53.4	49.7
efficientnet-b6-advprop-autoaug [19]	Robust	84.8	62.2	45.0	52.0	49.7
efficientnet-b7-advprop-autoaug [19]	Robust	85.1	62.0	44.7	53.2	49.6
efficientnet-b5-advprop-autoaug [19]	Robust	84.3	61.8	44.0	51.7	49.1
vit-large-patch16-224 [3]	More data	83.0	-	42.4	52.3	49.0
vit-base-patch32-384 [3]	More data	81.7	-	42.1	52.0	48.5
efficientnet-b4-advprop-autoaug [19]	Robust	82.7	59.2	42.5	51.4	48.1
resnet152 [7]	Standard	78.3	46.1	43.3	51.3	47.5
vit-base-patch16-224 [3]	More data	81.8	-	40.3	51.4	47.4
vit-large-patch32-384 [3]	More data	81.5	-	39.9	51.8	47.4
resnet101 [7]	Standard	77.4	44.1	40.5	50.1	46.3
efficientnet-b6-autoaug [1]	Standard	84.1	56.0	39.3	50.7	45.8
efficientnet-b5-autoaug [1]	Standard	83.6	55.7	39.8	50.2	45.7
efficientnet-b3-advprop-autoaug [19]	Robust	81.1	56.0	39.8	49.5	45.5
vit-base-patch8-224 [3]	More data	85.8	-	35.2	48.9	45.1
efficientnet-b7-autoaug [1]	Standard	84.3	54.7	39.1	49.9	45.0
efficientnet-b7-randaug [2]	Standard	84.7	63.1	38.9	48.7	45.0
efficientnet-b4-autoaug [1]	Standard	82.5	53.1	38.2	48.6	44.3
efficientnet-b2-advprop-autoaug [19]	Robust	79.6	51.0	38.2	48.0	44.3
resnet50 [7]	Standard	76.1	39.6	37.1	47.7	42.9
efficientnet-b5 [16]	Standard	83.1	53.5	37.0	47.3	42.8
efficientnet-b3-autoaug [1]	Standard	81.1	50.1	34.9	47.5	42.6
efficientnet-b1-advprop-autoaug [19]	Robust	78.5	50.5	36.2	46.7	42.4
resnet50-augmix [10]	Robust	77.5	48.3	35.9	46.4	42.2
efficientnet-b5-randaug [2]	Standard	83.5	57.3	35.5	47.3	42.1
efficientnet-b4 [16]	Standard	82.2	51.0	35.6	46.4	41.7
resnet101-lpf3 [22]	Robust	78.1	44.8	35.7	46.1	41.7
efficientnet-b2-autoaug [1]	Standard	79.8	49.0	34.3	45.8	41.6
resnet50-lpf5 [22]	Robust	77.0	41.6	35.2	45.3	41.5
resnet50-deepaugment [9]	Robust	76.7	52.9	34.9	46.0	41.3
resnet101-lpf2 [22]	Robust	77.8	44.4	35.1	45.2	41.1
resnet101-lpf5 [22]	Robust	77.9	44.7	34.8	45.8	41.0
efficientnet-b3 [16]	Standard	80.2	48.6	34.2	45.3	40.7
efficientnet-b0-advprop-autoaug [19]	Robust	77.1	47.3	34.2	44.9	40.5
resnet50-deepaugment-augmix [9]	Robust	75.8	56.7	34.1	44.5	40.3
resnet50-lpf2 [22]	Robust	76.8	40.6	33.4	45.1	40.3
resnet50-lpf3 [22]	Robust	76.8	41.3	34.3	44.7	40.0
efficientnet-b1-autoaug [1]	Standard	78.7	47.5	32.8	44.4	39.7
resnet26d [8]	Standard	76.7	-	35.8	43.5	39.7
resnet50-trained-on-SIN-IN-finetuned-on-IN [6]	Robust	76.7	42.0	32.4	44.6	39.2
resnet50-with-brightness-aws [17]	Robust	75.3	41.4	32.5	43.5	38.8
resnet50-with-gaussian-noise-contrast-motion-blur-jpeg-compression-aws [17]	Robust	72.7	46.8	33.6	42.9	38.8
densenet121-lpf5 [22]	Robust	75.0	39.7	32.0	42.7	38.7
resnet50-with-spatter-aws [17]	Robust	75.2	40.6	31.4	42.7	38.3

densenet121-lpf2 [22]	Robust	75.0	39.2	31.7	43.1	38.3
densenet121-lpf3 [22]	Robust	75.1	38.2	32.3	42.8	38.3
resnet34-lpf2 [22]	Robust	74.5	39.3	32.4	42.8	38.3
resnet34-lpf3 [22]	Robust	74.3	39.9	31.9	42.9	38.3
resnet50-with-saturate-aws [17]	Robust	74.9	39.3	32.4	42.4	38.2
resnet50-trained-on-SIN-and-IN [6]	Robust	74.6	45.3	32.5	42.7	38.2
efficientnet-b2 [16]	Standard	78.9	46.0	31.4	42.8	38.1
resnet50-with-pixelate-aws [17]	Robust	68.5	39.6	30.9	41.4	37.4
resnet34-lpf5 [22]	Robust	74.2	38.9	29.9	42.5	37.2
vgg16-bn-lpf2 [22]	Robust	74.0	34.1	31.3	41.8	37.2
vgg16-bn-lpf5 [22]	Robust	74.0	33.9	30.8	41.7	37.0
vgg16-bn-lpf3 [22]	Robust	73.9	34.5	30.6	42.1	36.9
efficientnet-b1 [16]	Standard	77.9	43.8	30.9	41.5	36.7
vgg16-bn [15]	Standard	73.4	34.0	31.1	41.1	36.7
resnet50-with-contrast-aws [17]	Robust	72.0	38.3	30.7	40.9	36.5
resnet50-with-jpeg-compression-aws [17]	Robust	73.6	39.8	30.3	41.3	36.5
resnet50-with-gaussian-noise-aws [17]	Robust	73.0	40.5	30.2	40.6	36.4
vgg19-bn [15]	Standard	74.2	35.7	29.4	40.8	36.2
resnet50-with-frost-aws [17]	Robust	73.8	39.8	29.7	40.0	36.1
mobilenet-v2-lpf3 [22]	Robust	72.6	33.1	30.4	40.3	36.0
resnet50-with-fog-aws [17]	Robust	71.8	35.4	30.3	39.9	35.9
mobilenet-v2-lpf5 [22]	Robust	72.5	33.3	29.1	40.1	35.8
resnet50-with-motion-blur-aws [17]	Robust	67.5	37.9	30.2	39.6	35.7
resnet18-lpf3 [22]	Robust	71.7	34.6	28.5	39.5	35.6
mobilenet-v2-lpf2 [22]	Robust	72.6	32.9	30.3	39.2	35.5
resnet18-lpf2 [22]	Robust	71.4	34.8	28.7	40.1	35.5
vgg16-lpf3 [22]	Robust	72.2	30.5	28.2	40.0	35.1
efficientnet-b0-autoaug [1]	Standard	76.8	41.8	27.3	40.1	34.9
resnet18-lpf5 [22]	Robust	71.4	34.9	27.7	38.9	34.7
vgg16 [15]	Standard	71.6	29.6	28.5	39.5	34.7
vgg19 [15]	Standard	72.4	30.5	29.0	39.3	34.7
vgg16-lpf5 [22]	Robust	72.3	30.2	27.8	39.4	34.5
efficientnet-b0 [16]	Standard	76.5	41.1	27.4	38.4	34.2
resnet18 [7]	Standard	69.8	33.1	27.7	39.1	34.2
vgg13-bn [15]	Standard	71.6	30.1	28.3	38.4	33.7
vgg16-lpf2 [22]	Robust	72.2	30.3	26.7	38.5	33.5
vgg11-bn [15]	Standard	70.4	29.9	25.8	37.1	32.9
resnet50-with-zoom-blur-aws [17]	Robust	61.3	32.5	28.3	36.6	32.7
vgg13 [15]	Standard	69.9	26.8	26.4	36.5	32.4
vgg11 [15]	Standard	69.0	27.0	25.2	36.1	31.5
resnet50-with-greyscale-aws [17]	Robust	63.3	24.2	24.3	34.3	30.2
resnet50-linf-eps4-robust [4]	Robust	62.4	32.2	19.5	31.9	27.2
resnet50-adv-train-free [14]	Robust	60.5	29.1	20.5	30.9	26.7
resnet50-trained-on-SIN [6]	Robust	60.2	37.7	20.4	30.0	25.3
resnet50-l2-eps3-robust [4]	Robust	57.9	31.0	18.4	29.4	24.8
alexnet-lpf3 [22]	Robust	56.9	21.5	17.5	26.8	23.1
alexnet-lpf2 [22]	Robust	57.2	21.3	18.2	26.8	22.8
alexnet-lpf5 [22]	Robust	56.6	21.5	18.4	26.8	22.7
resnet50-linf-eps8-robust [4]	Robust	47.9	24.2	16.0	25.7	21.6
resnet50-with-defocus-blur-aws [17]	Robust	31.9	18.2	12.6	19.3	16.2

Table 1. All 100 models were sorted based on their accuracy on all images of our dataset. IN and IN-C show the accuracy of models on ImageNet and ImageNet-C, respectively. Corr. shows the accuracy of models on all corrupted images of our dataset. The accuracy of the models on ImageNet-C is the average of the accuracy of the models on each corruption and severity.

[10] Dan Hendrycks, Norman Mu, Ekin Dogus Cubuk, Barret Zoph, Justin Gilmer, and Balaji Lakshminarayanan. Augmix: A simple data processing method to improve robustness and uncertainty. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. 6, 7

[11] Zhuang Liu, Hanzi Mao, Chaozheng Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11966–11976, 2022. 6, 7

[12] Dhruv Kumar Mahajan, Ross B. Girshick, Vignesh Ramanathan, Kaiming He, Manohar Paluri, Yixuan Li, Ashwin Bharambe, and Laurens van der Maaten. Exploring the limits of weakly supervised pretraining. In *ECCV*, 2018. 7

[13] T. Ridnik, Emanuel Ben-Baruch, Asaf Noy, and Lihi Zelnik-Manor. Imagenet-21k pretraining for the masses. *ArXiv*, abs/2104.10972, 2021. 6

[14] Ali Shafahi, Mahyar Najibi, Amin Ghiasi, Zheng Xu, John P. Dickerson, Christoph Studer, Larry S. Davis, Gavin Taylor, and Tom Goldstein. Adversarial training for free! In *NeurIPS*, 2019. 8

[15] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. 6, 8

[16] Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 6105–6114. PMLR, 2019. 6, 7, 8

[17] Rohan Taori, Achal Dave, Vaishaal Shankar, Nicholas Carlini, Benjamin Recht, and Ludwig Schmidt. Measuring robustness to natural distribution shifts in image classification. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 4, 6, 7, 8

[18] Ross Wightman. Pytorch image models. `https://github.com/rwightman/pytorch-image-models`, 2019. 4

[19] Cihang Xie, Mingxing Tan, Boqing Gong, Jiang Wang, Alan Loddon Yuille, and Quoc V. Le. Adversarial examples improve image recognition. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 816–825, 2020. 6, 7

[20] Saining Xie, Ross B. Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5987–5995, 2017. 6, 7

[21] Li Yuan, Qibin Hou, Zihang Jiang, Jiashi Feng, and Shuicheng Yan. Volo: Vision outlooker for visual recognition. *IEEE transactions on pattern analysis and machine intelligence*, PP, 2022. 6, 7

[22] Richard Zhang. Making convolutional networks shift-invariant again. In *ICML*, 2019. 6, 7, 8