ImageNet-E: Benchmarking Neural Network Robustness via Attribute Editing

Xiaodan Li\textsuperscript{1}, Yuefeng Chen\textsuperscript{1}\textsuperscript{*}, Yao Zhu\textsuperscript{2}, Shuhui Wang\textsuperscript{3}\textsuperscript{*}, Rong Zhang\textsuperscript{1}, Hui Xue\textsuperscript{1}
\textsuperscript{1}Alibaba Group  \textsuperscript{2}Zhejiang University  \textsuperscript{3}Inst. of Comput. Tech., CAS, China
\{fiona.lxd, yuefeng.chenyf, stone.zhangr, hui.xueh\}@alibaba-inc.com
\texttt{ee.zhuy@zju.edu.cn}, \texttt{wangshuhui@ict.ac.cn}

![ImageNet-C Examples](image1)

![ImageNet-E Examples](image2)

Figure 1. Examples of the proposed ImageNet-E dataset. In contrast to adversarial examples or datasets like ImageNet-C \cite{22} who add perturbation or corruptions to original images, we edit the object attributes with controls of backgrounds, sizes, positions and directions.

### Abstract

Recent studies have shown that higher accuracy on ImageNet usually leads to better robustness against different corruptions. Therefore, in this paper, instead of following the traditional research paradigm that investigates new out-of-distribution corruptions or perturbations deep models may encounter, we conduct model debugging in in-distribution data to explore which object attributes a model may be sensitive to. To achieve this goal, we create a toolkit for object editing with controls of backgrounds, sizes, positions, and directions, and create a rigorous benchmark named ImageNet-E(diting) for evaluating the image classifier robustness in terms of object attributes. With our ImageNet-E, we evaluate the performance of current deep learning models, including both convolutional neural networks and vision transformers. We find that most models are quite sensitive to attribute changes. A small change in the background can lead to an average of 9.23\% drop on top-1 accuracy. We also evaluate some robust models including both adversarially trained models and other robust trained models and find that some models show worse robustness against attribute changes than vanilla models. Based on these findings, we discover ways to enhance attribute robustness with preprocessing, architecture designs, and training strategies. We hope this work can provide some insights to the community and open up a new avenue for research in robust computer vision. The code and dataset are available at \url{https://github.com/alibaba/easyrobust}.

### 1. Introduction

Deep learning has triggered the rise of artificial intelligence and has become the workhorse of machine intelligence. Deep models have been widely applied in various fields such as autonomous driving \cite{28}, medical science \cite{33}, and finance \cite{38}. With the spread of these techniques, the robustness and safety issues begin to be essential, especially after the finding that deep models can be easily fooled by negligible noises \cite{16}. As a result, more researchers contribute to building datasets for benchmark-
ing model robustness to spot vulnerabilities in advance.

Most of the existing work builds datasets for evaluating the model robustness and generalization ability on out-of-distribution data [7, 22, 30] using adversarial examples and common corruptions. For example, the ImageNet-C (corruption) dataset conducts visual corruptions such as Gaussian noise to input images to simulate the possible processors in real scenarios [22]. ImageNet-R (enditions) contains various renditions (e.g., paintings, embroidery) of ImageNet object classes [21]. As both studies have found that higher accuracy on ImageNet usually leads to better robustness against different domains [22, 50]. However, most previous studies try to achieve this in a top-down way, such as architecture design, exploring a better training strategy, etc. We advocate that it is also essential to manage it in a bottom-up way, that is, conducting model debugging with the in-distribution dataset to provide clues for model repairing and accuracy improvement. It is interesting to explore whether a bird with a water background can be recognized correctly even if most birds appear with trees or grasses in the training data. Though this topic has been investigated in studies such as causal and effect analysis [9], the experiments and analysis are undertaken on domain generalization datasets. How a deep model generalizes to different backgrounds is still unknown due to the vacancy of a qualified benchmark. Therefore, in this paper, we provide a detached object editing tool to conduct the model debugging from the perspective of object attribute and construct a dataset named ImageNet-E (editing).

The ImageNet-E dataset is a compact but challenging test set for object recognition that contains controllable object attributes including backgrounds, sizes, positions and directions, as shown in Fig. 1. In contrast to ObjectNet [5] whose images are collected by their workers via posing objects according to specific instructions and differ from the target data distribution. This makes it hard to tell whether the degradation comes from the changes of attribute or distribution. Our ImageNet-E is automatically generated with our object attribute editing tool based on the original ImageNet. Specifically, to change the object background, we provide an object background editing method that can make the background simpler or more complex based on diffusion models [25, 46]. In this way, one can easily evaluate how much the background complexity can influence the model performance. To control the object size, position, and direction to simulate pictures taken from different distances and angles, an object editing method is also provided. With the editing toolkit, we apply it to the large-scale ImageNet dataset [42] to construct our ImageNet-E (editing) dataset. It can serve as a general dataset for benchmarking robustness evaluation on different object attributes.

With the ImageNet-E dataset, we evaluate the performance of current deep learning models, including both convolutional neural networks (CNNs), vision transformers as well as the large-scale pretrained CLIP [40]. We find that deep models are quite sensitive to object attributes. For example, when editing the background towards high complexity (see Fig. 1, the 3rd row in the background part), the drop in top-1 accuracy reaches 9.23% on average. We also find that though some robust models share similar top-1 accuracy on ImageNet, the robustness against different attributes may differ a lot. Meanwhile, some models, being robust under certain settings, even show worse results than the vanilla ones on our dataset. This suggests that improving robustness is still a challenging problem and the object attributes should be taken into account. Afterward, we discover ways to enhance robustness against object attribute changes. The main contributions are summarized as follows:

- We provide an object editing toolkit that can change the object attributes for manipulated image generation.

- We provide a new dataset called ImageNet-E that can be used for benchmarking robustness to different object attributes. It opens up new avenues for research in robust computer vision against object attributes.

- We conduct extensive experiments on ImageNet-E and find that models that have good robustness on adversarial examples and common corruptions may show poor performance on our dataset.

2. Related Work

The literature related to attribute robustness benchmarks can be broadly grouped into the following themes: robustness benchmarks and attribute editing datasets. Existing robustness benchmarks such as ImageNet-C (corruption) [22], ImageNet-R (enditions) [21], ImageNet-Stylized [14] and ImageNet-3DCC [30] mainly focus on the exploration of the corrupted or out-of-distribution data that models may encounter in reality. For instance, the ImageNet-R dataset contains various renditions (e.g., paintings, embroidery) of ImageNet object classes. ImageNet-C analyzes image models in terms of various simulated image corruptions (e.g., noise, blur, weather, JPEG compression, etc.). Attribute editing dataset creation is a new topic and few studies have explored it before. Among them, ObjectNet [5] and ImageNet-9 (a.k.a. background challenge) [50] can be the representative. Specifically, ObjectNet collects a large real-world test set for object recognition with controls where object backgrounds, rotations, and imaging viewpoints are random. The images in ObjectNet are collected by their workers who image objects in their homes. It consists of 313 classes which are mainly household objects. ImageNet-9 mainly creates a suit of datasets that help disentangle the impact of foreground and background signals on classification. To achieve this goal, it uses coarse-grained classes with corresponding rectangular bounding boxes to remove
3. Preliminaries

Since the editing tool is developed based on diffusion models, let us first briefly review the theory of denoising diffusion probabilistic models (DDPM) [25,46] and analyze how it can be used to generate images.

According to the definition of the Markov Chain, one can always reach a desired stationery distribution from a given distribution along with the Markov Chain [15]. To get a generative model that can generate images from random Gaussian noises, one only needs to construct a Markov Chain whose stationary distribution is Gaussian distribution. This is the core idea of DDPM. In DDPM, given a data distribution \( x_0 \sim q(x_0) \), a forward noising process produces a series of latents \( x_1, ..., x_T \) of the same dimensionality as the data \( x_0 \) by adding Gaussian noise with variance \( \beta_t \in (0, 1) \) at time \( t \):

\[
q(x_t|x_{t-1}) = N(\sqrt{1-\beta_t}x_{t-1}, \beta_t I), \ s.t. \ 0 < \beta_t < 1,
\]

where \( \beta_t \) is the diffusion rate. Then the distribution \( q(x_t|x_0) \) at any time \( t \) is:

\[
q(x_t|x_0) = N(\sqrt{\alpha_t}(1-\alpha_t)I, \ x_t = \sqrt{\alpha_t}x_0 + \sqrt{1-\alpha_t}\epsilon)
\]

(2)

where \( \alpha_t = \prod_{s=1}^{t}(1-\beta_s) \), \( \epsilon \sim N(0, I) \). It can be proved that \( \lim_{t\to\infty} q(x_t) = N(0, I) \). In other words, we can map the original data distribution into a Gaussian distribution with enough iterations. Such a stochastic forward process is named as diffusion process since what the process \( q(x_t|x_{t-1}) \) does is adding noise to \( x_{t-1} \).

To draw a fresh sample from the distribution \( q(x_0) \), the Markov process is reversed. That is, beginning from a Gaussian noise sample \( x_T \sim N(0, I) \), a reverse sequence is constructed by sampling the posteriors \( q(x_{t-1}|x_t) \). To approximate the unknown function \( q(x_{t-1}|x_t) \), in DDPMs, a deep model \( p\theta \) is trained to predict the mean and the covariance of \( x_{t-1} \) given \( x_t \) instead. Then the \( x_{t-1} \) can be sampled from the normal distribution defined as:

\[
p\theta(x_{t-1}|x_t) = N(\mu\theta(x_t, t), \Sigma\theta(x_t, t)).
\]

(3)

In stead of inferring \( \mu\theta(x_t, t) \) directly, [25] propose to predict the noise \( \epsilon\theta(x_t, t) \) which was added to \( x_0 \) to get \( x_t \) with Eq. (2). Then \( \mu\theta(x_t, t) \) is:

\[
\mu\theta(x_t, t) = \frac{1}{\sqrt{\alpha_t}} \left( x_t - \frac{\beta_t}{\sqrt{1-\alpha_t}}\epsilon\theta(x_t, t) \right).
\]

(4)

[25] keep the value of \( \Sigma\theta(x_t, t) \) to be constant. As a result, given a sample \( x_t \) at time \( t \), with a trained model that can predict the noise \( \epsilon\theta(x_t, t) \), we can get \( \mu\theta(x_t, t) \) according to Eq. (4) to reach the \( x_{t-1} \) with Equation (3) and eventually we can get to \( x_0 \).

Previous studies have shown that diffusion models can achieve superior image generation quality compared to the current state-of-the-art generative models [1]. Besides, there have been plenty of works on utilizing the DDPMs to generate samples with desired properties, such as semantic image translation [37], high fidelity data generation from low-density regions [45], etc. In this paper, we also choose the DDPM adopted in [1] as our generator.

4. Attribute Editing with Diffusion Models and ImageNet-E

Most previous robustness-related work has focused on the important challenges of robustness on adversarial examples [7], common corruptions [22]. They have found that higher clean accuracy usually leads to better robustness. Therefore, instead of exploring a new corruption that models may encounter in reality, we pay attention to the model debugging in terms of object attributes, hoping to provide new insights to clean accuracy improvement. In the following, we describe our object attribute editing tool and the generated ImageNet-E dataset in detail.

4.1. Object Attribute Editing with Diffusion Models

Background editing. Most existing corruptions conduct manipulations on the whole image, as shown in Fig. 1. Compared to adding global corruptions that may hinder the visual quality, a more likely-to-happen way in reality is to manipulate the backgrounds to fool the model. Besides, it is shown that there exists a spurious correlation between labels and image backgrounds [13]. From this point, a background corruption benchmark is needed to evaluate the model’s robustness. However, the existing background challenge dataset achieves background editing with copy-paste operation, resulting an obvious artifacts in generated images [50]. This may leave some doubts about whether the evaluation is precise since the dataset’s distribution may have changed. To alleviate this concern, we adopt DDPM approach to incorporate background editing by adding a guiding loss that can lead to backgrounds with desired properties to make the generated images stay close to the original distribution. Specifically, we choose to manipulate the background in terms of texture complexity due to the hypothesis that an object should be observed more easily from simple backgrounds than from complicated ones. In general, the texture complexity can be evaluated with the gray-level co-occurrence matrix (GLCM) [17], which calculates the gray-level histogram to show the texture characteristic. However, the calculation of GLCM is non-differentiable, thus it cannot serve as the conditional guidance of image generation. We hypothesize that a complex image should contain more frequency components in its spectrum and higher amplitude
indicates greater complexity. Thus, we define the objective of complexity as:

\[ L_c = \sum |A(\mathcal{F}(x))|, \]

(5)

where \( \mathcal{F} \) is the Fourier transform [6], \( A \) extracts the amplitude of the input spectrum. \( x \) is the evaluated image. Since minimizing this loss helps us generate an image with desired properties and should be conducted on the ground image \( x_0 \), we need a way of estimating a clean image \( x_0 \) from each noisy latent representation \( x_t \) during the denoising diffusion process. Recall that the process estimates at each step the noise \( \epsilon_\theta(x_t, t) \) added to \( x_0 \) to obtain \( x_t \). Thus, \( x_0 \) can be estimated via Equation (6) [1]. The whole optimization procedure is shown in Algorithm 1.

\[ \hat{x}_0 = \frac{x_t}{\sqrt{\alpha_t}} - \sqrt{1 - \alpha_t} \epsilon_\theta(x_t, t) \]

(6)

As shown in Fig. 2(a), with the proposed method, when we guide the generation procedure with the proposed objective towards the complex direction, it will return images with visually complex backgrounds. We also provide the GLCM dissimilarity and contrast of each image to make a quantitative analysis of the generated images. A higher dissimilarity/contrast score indicates a more complex image background [17]. It can be observed that the complexity is consistent with that calculated with GLCM, indicating the effectiveness of the proposed method.

**Controlling object size, position and direction.** In general, the human vision system is robust to position, direction and small size changes. Whether the deep models are also robust to these object attribute changes is still unknown to researchers. Therefore, we conduct the image editing with controls of object sizes, positions and directions to find the answer. For a valid evaluation on different attributes, all other variables should remain unchanged, especially the background. Therefore, we first disentangle the object and background with the in-painting strategy provided by [54]. Specifically, we mask the object area in input image \( x \). Then we conduct in-painting to remove the object and get the pure background image \( x^b \), as shown in Fig. 2(b) column 3. To realize the aforementioned object attribute controlling, we adopt the orthogonal transformation. Denote \( P \) as the pixel locations of object in image \( x \) where \( P \in \mathbb{R}^{N \times 2} \). \( N \) is the number of pixels belong to object and \( p_i = [x_i, y_i, 1]^T \) is the position of object’s i-th pixel. \( h' \in [0, H - h], w' \in [0, W - w] \) where \( [x, y, w, h] \) stand for the enclosing rectangle of the object with mask \( M \). Then the newly edited \( x[T_{\text{attribute}} \cdot P] = x[P] \) and \( M[T_{\text{attribute}} \cdot P] = M[P] \), where

\[
T_{\text{size}} = \begin{bmatrix}
0 & s \Delta x \\
0 & s \Delta y \\
0 & 1
\end{bmatrix},
T_{\text{position}} = \begin{bmatrix}
1 & 0 & w' \\
0 & 1 & h' \\
0 & 0 & 1
\end{bmatrix},
T_{\text{direction}} = \begin{bmatrix}
\cos \theta & \sin \theta & 0 \\
-\sin \theta & \cos \theta & 0 \\
0 & 0 & 1
\end{bmatrix}.
\]

(7)

where \( s \) is the resize scale, \( \theta \) is the rotation angle. \( \Delta x = (1 - s) \cdot (x + w/2), \Delta y = (1 - s) \cdot (y + h/2) \).

With the background image \( x^b \) and edited object \( x^o \), a naive way is to place the object in the original image to the corresponding area of background image \( x^b \) as \( M \odot x^o + (1 - M) \odot x^b \). However, the result generated in this manner may look disharmonic, lacking a delicate adjustment to blending them together. Besides, as shown in Fig. 2(b) column 3, the object-removing operation may leave some artifacts behind, failing to produce a coherent and seamless result. To deal with this problem, we leverage DDPM models to blend them at different noise levels along the diffusion process. Denote the image with desired object attribute as \( x^o \). Starting from the pure background image \( x^b \) at time \( t_0 \), at each stage, we perform a guided diffusion step with a latent \( x_t \) to obtain the \( x_{t-1} \) and at the same time, obtain a noised version of object image \( x^o_{t-1} \). Then the two latents are blended with the mask \( M \) as \( x_{t-1} = M \odot x^o_{t-1} + (1 - M) \odot x^b_{t-1} \). The DDPM denoising procedure may change the background. Thus a proper initial timing is required to maintain a high resemblance to the original background. We set the iteration steps \( t_0 \) as 50 and 25 in Algorithm 1 and 2 respectively.
Algorithm 1: Background editing

input : source image \( x \), mask \( M \), diffusion model \( (\mu_0(x), \Sigma_0(x)) \), \( \alpha_t \), \( \lambda \), iteration steps \( t_0 \)
output: edited image \( x_0 \)

\[
1 \quad x_{t_0} \sim \mathcal{N}(\sqrt{\alpha_{t_0}}x, (1 - \alpha_{t_0})I);
2 \quad \text{for } t \leftarrow t_0 \text{ to } 1 \text{ do}
3 \quad x_0 \leftarrow \frac{x}{\sqrt{\bar{t}}}; \quad \sqrt{\bar{t}}_t \mu_t(x); \quad \nabla_b \leftarrow \nabla_b \mathcal{L}_c(x_t);
4 \quad x_{t-1} \sim \mathcal{N}(\mu_t(x_t) + \lambda \Sigma_0(x_t) \nabla_b, \Sigma_0(x_t));
5 \quad x^t \sim \mathcal{N}(\sqrt{\bar{t}}_t x, (1 - \bar{t}_t)I);
6 \quad x_{t-1} \leftarrow M \odot x^t + (1 - M) \odot x_{t-1};
7 \quad \text{end}
\]

Algorithm 2: Object size controlling

input : source image \( x \), mask \( M \), diffusion model \( (\mu_0(x), \Sigma_0(x)) \), \( \alpha_t \), iteration steps \( t_0 \), ratio \( s \)
output: edited image \( x_0 \)

\[
1 \quad \hat{x}^t \leftarrow \text{ObjectRemoving}(x, M);
2 \quad x, M \leftarrow \text{Rescale}(x, M, s);
3 \quad x_{t_0} \sim \mathcal{N}(\sqrt{\alpha_{t_0}}x^b, (1 - \alpha_{t_0})I);
4 \quad \text{for } t \leftarrow t_0 \text{ to } 1 \text{ do}
5 \quad x_{t-1} \sim \mathcal{N}(\mu_t(x_t), \Sigma_0(x_t));
6 \quad x^o \sim \mathcal{N}(\sqrt{\bar{t}}_t x^b, (1 - \bar{t}_t)I);
7 \quad x_{t-1} \leftarrow M \odot x^o + (1 - M) \odot x_{t-1};
8 \quad \text{end}
\]

4.2. ImageNet-E dataset

With the tool above, we conduct object attribute editing including background, size, direction and position changes based on the large-scale ImageNet dataset [42] and ImageNet-S [12], which provides the mask annotation. To guarantee the dataset quality, we choose the animal classes from ImageNet classes such as dogs, fishes and birds, since they appear more in nature without messy backgrounds. Classes such as stove and mortarboard are removed. Finally, our dataset consists of 47872 images with 373 classes based on the initial 4352 images, each of which is applied 11 transforms. Detailed information can be found in Appendix A. For background editing, we choose three levels of the complexity, including \( \lambda = -20, \lambda = 20 \) and \( \lambda = 20 \)-adv with adversarial guidance (see Sec.B for details) instead of complexity. Larger \( \lambda \) indicates stronger guidance towards high complexity. For the object size, we design four levels of sizes in terms of the object pixel rates (= \( \sum(M > 0.5)/\sum(M \geq 0) \)): [Full, 0.1, 0.08, 0.05] where ‘Full’ indicates making the object as large as possible while maintaining its whole body inside the image. Smaller ratios indicate smaller objects. For object position, we find that some objects hold a high object pixel rate in the whole image, resulting in a small \( H - h \). Take the first picture in Fig. 2 for example, the dog is big and it will make little visual differences after position changing. Thus, we adopt the data whose pixel rate is 0.05 as the initial images for the position-changing operation.

In contrast to benchmarks like ImageNet-C [22] giving images from different domains so that the model robustness in these situations may be assessed, our effort aims to give an editable image tool that can conduct model debugging with in-distribution (ID) data, in order to identify specific shortcomings of different models and provide some insights for clean accuracy improving. Thus, the data distribution should not differ much from the original ImageNet. We choose the out-of-distribution (OOD) detection methods Energy [34] and GradNorm [27] to evaluate whether our editing tool will move the edited image out of its original distribution. These OOD detection methods aim to distinguish the OOD examples from the ID examples. The results are shown in Fig. 3. \( x \)-axis is the ID score in terms of the quantities in Energy and GradNorm and \( y \)-axis is the frequency of each ID score. A high ID score indicates the detection method takes the input sample as the ID data. Compared to other datasets, our method barely changes the data distribution under both Energy (the 1st row) and GradNorm (the 2nd row) evaluation methods. Besides, the Fréchet inception distance (FID) [24] for our ImageNet-E is 15.57 under the random background setting, while it is 34.99 for ImageNet-9 (background challenge). These all imply that our editing tool can ensure the proximity to the original ImageNet, thus can give a controlled evaluation on object attribute changes. To find out whether the DDPM will induce some degradation to our evaluation, we have conducted experiment in Tab. 1 with the setting \( \lambda = 0 \) during background editing. This operation will first add noises to the original and then denoise them. It can be found in “Inver” column that the degradation is negligible compared to degradation induced by attribute changes.

5. Experiments

We conduct evaluation experiments on various architectures including both CNNs (ResNet (RN) [20], DenseNet [26], EfficientNet (EF) [47], ResNet [53], ConvNeXt [36]) and transformer-based models (Vision-Transformer (ViT) [10], Swin-Transformer (Swin) [35]). Other state-of-the-art models that trained with extra data such as CLIP [40], EfficientNet-L2-Noisy-Student [51] are also evaluated in the Appendix. Apart from different sizes of these models, we have also evaluated their adversarially trained versions for comprehensive studies. We report the drop of top-1 accuracy as metric based on the idea that the attribute changes should induce little influence to a robust trained model. More experimental details and results of top-1 accuracy can be found in the Appendix.
5.1. Robustness evaluation

Normally trained models. To find out whether the widely used models in computer vision have gained robustness against changes on different object attributes, we conduct extensive experiments on different models. As shown in Tab. 1, when only the background is edited towards high complexity, the average drop in top-1 accuracy is 9.23% ($\lambda = 20$). This indicates that most models are sensitive to object background changes. Other attribute changes such as size and position can also lead to model performance degradation. For example, when changing the object pixel rate to 0.05, as shown in Fig. 1 row 4 in the ‘size’ column, while we can still recognize the image correctly, the performance drop is 18.34% on average. We also find that the robustness under different object attributes is improved along with improvements in terms of clean accuracy (Original) on different models. Accordingly, a switch from an RN50 (92.69% top-1 accuracy) to a Swin-S (96.21%) leads to the drop in accuracy decrease from 15.72% to 10.20% on average. By this measure, models have become more and more capable of generalizing to different backgrounds, which implies that they indeed learn some robust features. This shows that object attribute robustness can be a good way to measure future progress in representation learning. We also observe that larger networks possess better robustness on the attribute editing. For example, swapping a Swin-S (96.21% top-1 accuracy) with the larger Swin-B (95.96% top-1 accuracy) leads to the decrease of the dropped accuracy from 10.20% to 8.99% when $\lambda = 20$. In a similar fashion, a ConvNeXt-T (9.32% drop) is less robust than the giant ConvNeXt-B (7.26%). Consequently, models with even more depth, width, and feature aggregation may attain further attribute robustness. Previous studies [31] have shown that zero-shot CLIP exhibits better out-of-distribution robustness than the finetuned CLIP, which is opposite to our ImageNet-E as shown in Tab. 1. This may serve as the evidence that our ImageNet-E has a good proximity to ImageNet. We also find that compared with fully-supervised trained model under the same backbone (ViT-B), the CLIP fails to show a better attribute robustness. We think this may be caused by that the CLIP has spared some capacity for OOD robustness.

Adversarially trained models. Adversarial training [43] is one of the state-of-the-art methods for improving the adversarial robustness of deep models and has been widely studied [2]. To find out whether they can boost the attribute robustness, we conduct extensive experiments in terms of different architectures and perturbation budgets (constraints of $l_2$ norm bound). As shown in Fig. 4, the adversarially trained ones are not robust against attribute changes including both backgrounds and size-changing situations. The dropped accuracies are much greater compared to normally trained models. As the perturbation budget grows, the situation gets worse. This indicates that adversarial training can do harm to robustness against attributes.

5.2. Robustness enhancements

Based on the above evaluations, we step further to discover ways to enhance the attribute robustness in terms of preprocessing, network design and training strategies. More details including training setting and numerical experimental results can be found in Appendix C.5.

Preprocessing. Given that an object can be inconspicuous due to its small size or subtle position, viewing an object at
several different locations may lead to a more stable prediction. Having this intuition in mind, we perform the classical Ten-Crop strategy to find out if this operation can help to get a robustness boost. The Ten-Crop operation is executed by cropping all four corners and the center of the input image. We average the predictions of these crops together with their horizontal mirrors as the final result. We find this operation can contribute a 0.69% and 1.24% performance boost on top-1 accuracy in both background and size changes scenarios on average respectively.

**Network designs.** Intuitively, a robust model should tend to focus more on the object of interest instead of the background. Therefore, recent models begin to enhance the model by employing attention modules. Of these, the ResNet [53] can be a representative. The ResNet is a modularized architecture, which applies channel-wise attention on different network branches to leverage their success in capturing cross-feature interactions and learning diverse representations. As it has achieved a great boost in the ImageNet dataset, it also shows superiority on ImageNet-E compared to ResNet. For example, a switch from RN50 decreases the average dropped accuracy from 15.72% to 12.57%. This indicates that the channel-wise attention module can be a good choice to improve the attribute robustness. Another representative model can be the vision transformer, which consists of multiple self-attention modules. To study whether incorporating transformer’s self-attention-like architecture into the model design can help attribute robustness generalization, we establish a hybrid architecture by directly feeding the output of res_3 block in RN50 into ViT-S as the input feature like [3]. The dropped accuracy decreases by 1.04% compared to the original RN50, indicating the effectiveness of the self-attention-like architectures.

**Training strategy.** a) **Robust trained.** There have been plenty of studies focusing on the robust training strategy to improve model robustness. To find out whether these works can boost the robustness on our dataset, we further evaluate these state-of-the-art models including SIN [14], Debiased-CNN [32], Augmix [23], ANT [41], DeepAugment [21] and model trained with lots of standard augmentations (RN50-T) [48]. As shown in Tab. 2, apart from the RN50-T, while the Augmix model shows the best performance against the background change scenario, the Debiased model holds the best in the object size change scenario. What we find unexpectedly is the SIN performance. The SIN method features the novel data augmentation scheme where ImageNet images are stylized with style transfer as the training data to force the model to rely less on textural cues for classification. Though the robustness boost is achieved on ImageNet-C (mCE 69.32%) compared to its vanilla model (mCE 76.7%), it fails to improve the robustness in both object background and size-changing scenarios. The drops of top-1 accuracy for vanilla RN50 and RN50-SIN are 21.26% and 24.23% respectively, when the object size rate is 0.05, though they share similar accuracy on original ImageNet. This indicates that existing benchmarks cannot reflect the real robustness in object attribute changing. Therefore, a dataset like ImageNet-E is necessary for comprehensive evaluations on deep models. b) **Masked image modeling.** Considering that masked image modeling has demonstrated impressive results in self-supervised representation learning by recovering corrupted image patches [4], it may be robust to the attribute changes. Therefore, we choose the Masked AutoEncoder (MAE) [18] as the training strategy since its objective is recovering images with only 25% patches. Specifically, we adopt the MAE training strategy with ViT-B backbone and then finetune it with ImageNet training data. We find that the robustness is improved. For example, the dropped accuracy decreases from 10.62% to 9.05% on average compared to vanilla ViT-B.

<table>
<thead>
<tr>
<th>Models</th>
<th>Original</th>
<th>Background changes</th>
<th>Size changes</th>
<th>Position</th>
<th>Direction</th>
<th>Avg.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Inver λ = -20</td>
<td>λ = 20</td>
<td>λ = 20-adv</td>
<td>Random</td>
<td>Full</td>
<td>0.1</td>
</tr>
<tr>
<td>RN50</td>
<td>92.69%</td>
<td>91.79%</td>
<td>91.08%</td>
<td>92.92%</td>
<td>13.34%</td>
<td>2.71%</td>
</tr>
<tr>
<td>DenseNet121</td>
<td>92.10%</td>
<td>91.71%</td>
<td>91.02%</td>
<td>92.90%</td>
<td>12.43%</td>
<td>3.03%</td>
</tr>
<tr>
<td>EF-B</td>
<td>92.85%</td>
<td>91.70%</td>
<td>91.01%</td>
<td>92.88%</td>
<td>15.64%</td>
<td>2.53%</td>
</tr>
<tr>
<td>ResNet50</td>
<td>95.38%</td>
<td>94.71%</td>
<td>94.21%</td>
<td>94.89%</td>
<td>11.28%</td>
<td>0.80%</td>
</tr>
<tr>
<td>ViT-S</td>
<td>94.14%</td>
<td>92.62%</td>
<td>92.12%</td>
<td>92.88%</td>
<td>13.06%</td>
<td>1.13%</td>
</tr>
<tr>
<td>Swin-S</td>
<td>96.21%</td>
<td>95.61%</td>
<td>95.11%</td>
<td>95.76%</td>
<td>9.31%</td>
<td>1.27%</td>
</tr>
<tr>
<td>ConvNeXt-T</td>
<td>96.07%</td>
<td>95.47%</td>
<td>94.97%</td>
<td>95.16%</td>
<td>7.93%</td>
<td>1.75%</td>
</tr>
<tr>
<td>RN101</td>
<td>94.00%</td>
<td>93.41%</td>
<td>92.91%</td>
<td>93.41%</td>
<td>13.57%</td>
<td>2.57%</td>
</tr>
<tr>
<td>DenseNet169</td>
<td>92.37%</td>
<td>91.77%</td>
<td>91.27%</td>
<td>91.77%</td>
<td>11.61%</td>
<td>2.25%</td>
</tr>
<tr>
<td>EF-B3</td>
<td>94.97%</td>
<td>94.29%</td>
<td>93.79%</td>
<td>94.29%</td>
<td>12.92%</td>
<td>1.36%</td>
</tr>
<tr>
<td>ResNet101</td>
<td>95.54%</td>
<td>94.85%</td>
<td>94.35%</td>
<td>94.85%</td>
<td>10.40%</td>
<td>1.35%</td>
</tr>
<tr>
<td>ViT-B</td>
<td>95.38%</td>
<td>94.69%</td>
<td>94.19%</td>
<td>94.69%</td>
<td>10.98%</td>
<td>0.62%</td>
</tr>
<tr>
<td>Swin-B</td>
<td>95.96%</td>
<td>95.26%</td>
<td>94.76%</td>
<td>95.26%</td>
<td>8.25%</td>
<td>0.99%</td>
</tr>
<tr>
<td>ConvNeXt-B</td>
<td>96.42%</td>
<td>95.72%</td>
<td>95.22%</td>
<td>95.72%</td>
<td>6.04%</td>
<td>0.99%</td>
</tr>
<tr>
<td>CLIP-zero-shot</td>
<td>80.01%</td>
<td>78.31%</td>
<td>76.61%</td>
<td>78.31%</td>
<td>20.09%</td>
<td>3.33%</td>
</tr>
<tr>
<td>CLIP-finetuned</td>
<td>93.68%</td>
<td>92.98%</td>
<td>92.48%</td>
<td>92.98%</td>
<td>18.19%</td>
<td>9.06%</td>
</tr>
</tbody>
</table>
able in ImageNet. We also step further to find out if the only 8781 training images with mask annotations are avail-

77.48% to 79.00%. Note that the promotion is not small as ers during training and get a validation accuracy boost from we randomly replace the backgrounds of objects with oth-

ing. Previous evaluation shows that the ResNet50 is vul-

enhancement, we conduct a toy example for model repair-

5.4. Model repairing

To validate that the evaluation on ImageNet (IN)-E can help to provide some insights for model’s applicability and enhancement, we conduct a toy example for model repair-

5.3. Failure case analysis

To explore the reason why some robust trained mod-

eas from both preprocessing, network designing and training strategies.

Limitations and future work. This paper proposes to edit the object attributes in terms of backgrounds, sizes, positions and directions. Therefore, the annotated mask of the interest object is required, resulting in a limitation of our method. Besides, since our editing toolkit is developed based on diffusion models, the generalization ability is determined by DDPMs. For example, we find synthesizing high-quality person images is difficult for DDPMs. Under the consideration of both the annotated mask and data quality, our ImageNet-E is a compact test set. In our future work, we would like to explore how to leverage the edited data to enhance the model’s performance, including both the validation accuracy and robustness.

Table 3. Model repairing results. Top-1 accuracy (%) is reported except for IN-C, which is mCE (mean Corruption Error). Higher top-1 accuracy and lower mCE indicate better performance. IN-E reports the average accuracy on ImageNet-E.

<table>
<thead>
<tr>
<th>Models</th>
<th>IN</th>
<th>IN-v2</th>
<th>IN-A</th>
<th>IN-C</th>
<th>IN-R</th>
<th>IN-Sketch</th>
<th>IN-E</th>
</tr>
</thead>
<tbody>
<tr>
<td>RN50</td>
<td>77.3</td>
<td>65.7</td>
<td>6.5</td>
<td>68.6</td>
<td>39.6</td>
<td>27.5</td>
<td>83.7</td>
</tr>
<tr>
<td>RN50-repaired</td>
<td>79.0</td>
<td>67.2</td>
<td>9.4</td>
<td>65.8</td>
<td>40.7</td>
<td>29.4</td>
<td>85.0</td>
</tr>
</tbody>
</table>

Table 2. Evaluations with different robust models in terms of Top-1 accuracy and the corresponding dropped accuracy.

<table>
<thead>
<tr>
<th>Architectures</th>
<th>Ori</th>
<th>Background changes</th>
<th>Size changes</th>
<th>Position</th>
<th>Direction</th>
<th>Avg.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Inver</td>
<td>Random</td>
<td>Full</td>
<td>0.1</td>
<td>0.038</td>
</tr>
<tr>
<td>RN50</td>
<td>92.69%</td>
<td>0.76%</td>
<td>7.30%</td>
<td>0.05</td>
<td>0.40</td>
<td>0.76</td>
</tr>
<tr>
<td>RN50-Adversarial</td>
<td>81.96%</td>
<td>0.66%</td>
<td>4.75%</td>
<td>13.62%</td>
<td>37.87%</td>
<td>15.25%</td>
</tr>
<tr>
<td>RN50-SIN</td>
<td>91.57%</td>
<td>2.23%</td>
<td>7.61%</td>
<td>12.19%</td>
<td>33.16%</td>
<td>13.58%</td>
</tr>
<tr>
<td>RN50-Debiased</td>
<td>93.34%</td>
<td>1.43%</td>
<td>6.09%</td>
<td>11.45%</td>
<td>27.99%</td>
<td>12.12%</td>
</tr>
<tr>
<td>RN50-Augmix</td>
<td>93.50%</td>
<td>0.98%</td>
<td>6.26%</td>
<td>8.38%</td>
<td>30.49%</td>
<td>12.94%</td>
</tr>
<tr>
<td>RN50-ANT</td>
<td>91.87%</td>
<td>1.68%</td>
<td>6.62%</td>
<td>11.94%</td>
<td>35.66%</td>
<td>15.36%</td>
</tr>
<tr>
<td>RN50-DeepAugment</td>
<td>92.88%</td>
<td>1.50%</td>
<td>6.62%</td>
<td>12.37%</td>
<td>32.40%</td>
<td>13.32%</td>
</tr>
<tr>
<td>RN50-T</td>
<td>94.55%</td>
<td>1.05%</td>
<td>5.65%</td>
<td>7.38%</td>
<td>21.89%</td>
<td>10.42%</td>
</tr>
</tbody>
</table>

Figure 5. Heat maps for explaining which parts of the image dom-

inates the model decision through LayerCAM [29].

Figure 6. Image editing toolkit.

20378
References


[18] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 16000–16009, 2022. 7

[19] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 16000–16009, 2022. 15


[46] Zhenda Xie, Zheng Zhang, Yue Cao, Yutong Lin, Jianmin Bao, Zhuliang Yao, Qi Dai, and Han Hu. Simmim: A simple

