# Spurious Features Everywhere - Large-Scale Detection of Harmful Spurious Features in ImageNet

Yannic Neuhaus

Maximilian Augustin Valentyn Boreiko Tübingen AI Center – University of Tübingen Matthias Hein

## Abstract

Benchmark performance of deep learning classifiers alone is not a reliable predictor for the performance of a deployed model. In particular, if the image classifier has picked up spurious features in the training data, its predictions can fail in unexpected ways. In this paper, we develop a framework that allows us to systematically identify spurious features in large datasets like ImageNet. It is based on our neural PCA components and their visualization. Previous work on spurious features often operates in toy settings or requires costly pixel-wise annotations. In contrast, we work with ImageNet and validate our results by showing that presence of the harmful spurious feature of a class alone is sufficient to trigger the prediction of that class. We introduce the novel dataset "Spurious ImageNet" which allows to measure the reliance of any ImageNet classifier on harmful spurious features. Moreover, we introduce SpuFix as a simple mitigation method to reduce the dependence of any ImageNet classifier on previously identified harmful spurious features without requiring additional labels or retraining of the model. We provide code and data at https://github.com/YanNeu/spurious\_imagenet.

#### 1. Introduction

Deep learning has led to tremendous progress in image classification [37, 50] and natural language processing [14]. Over the years, however, it has become apparent, that evaluating predictive performance on a fixed test set is not necessarily indicative of the performance when image classifiers are deployed in the wild. Several potential failure cases have been discovered. This starts with a lack of robustness due to image corruptions [31], adversarial perturbations [68], and arbitrary predictions on out-of-distribution inputs [45, 32, 30]. In this paper, we consider the problem of identifying and debugging image classifiers from spurious features [2]. Spurious features in image classification are features that co-occur with the actual class object and are picked up by the classifier. In the worst case, they lead to shortcut learning [25], where only the spurious but not the



Figure 1: **Top:** Examples of spurious features found via our neural PCA components but not in previous study [61]. **Bottom:** We validate our spurious features by mining images from the web showing **only the spurious feature but not the class.** They are classified by four ImageNet models as the corresponding class. Some of them even contain ImageNet classes (bees on feeder, grasshopper in leaves).

correct feature is associated with the class, e.g., [86] found that a pneumonia detector's bad generalization across hospitals was caused by the neural network learning to identify the hospital where the training data originated from. A weaker form of spurious feature (at least from a learning perspective) is the case when the classifier picks up the correct class features, e.g., of a hummingbird, but additionally associates a spurious feature, e.g., a bird feeder, with the class as they appear together on a subset of the training set. This becomes a harmful spurious feature if *only* the spurious feature *without* the class feature is sufficient to trigger the classification of that class, see Fig. 1 for an illustration of such spurious features found via our method. Harmful spurious features are difficult to detect and thus can easily go unnoticed, leading to unexpected behaviour of a deployed image classifier.

In this paper we make the following key contributions:

- we develop a pipeline for the detection of harmful spurious features with little human supervision based on our class-wise neural PCA (NPCA) components of an adversarially robust classifier together with their Neural PCA Feature Visualization (NPFV).
- unlike prior work, which used masking images or pixelwise annotations, we validate our found spurious features by using our NPCA components to find real images containing only the spurious feature but not the class object.
- using these images we create the dataset "Spurious ImageNet" and propose a measure for dependence on spurious features. We do a large-scale evaluation of stateof-the-art (SOTA) ImageNet models. We show that the spurious features found for the robust model generalize to non-robust classifiers. Moreover, we analyze the influence of different training setups, e.g. pre-training on ImageNet21k or larger datasets like LAION.
- we develop SpuFix, a technique to mitigate the dependence on identified harmful spurious features without requiring new labels or retraining, and show how to transfer it to any ImageNet classifier. SpuFix consistently improves the dependence on harmful spurious features even for SOTA models with negligible impact on test accuracy.

#### 2. Related work

When classifiers in safety-critical systems such as healthcare or autonomous driving are deployed in the wild [3], it is important to discover potential failure cases before release. Prior work has focused on corruption [31], adversarial robustness [11, 68, 43], and out-of-distribution detection [45, 32, 30]. There is less work on spurious features, although their potential harm might be higher.

**Spurious features:** It has been noted early on that classifiers show reliance on spurious features [15] e.g., a cow on the beach is not recognized [9] due to the missing spurious feature of grass. Other forms of spurious features have been reported in the classification of skin lesions [12], pneumonia [86], traffic signs [67], and object recognition [88]. Moreover, it has been shown that deep neural networks are biased towards texture [26] and background context [80], see [25] for an overview. [58] argues theoretically that spurious features are picked up due to a simplicity bias.

Detection of spurious features has been achieved using human label-intense pixel-wise annotations [48, 59, 60]. In [78], they use sparsity regularization to enforce a more interpretable model and use it for finding spurious features. [4] propose a complex pipeline to detect spurious features. While they scale to ImageNet, their analysis is limited to a few spurious features for a subset of 100 classes. [61, 44, 62] do a search on full ImageNet based on classweighted "neural maps". The neural maps are used to add noise to "spurious" resp. "core" features but no significant difference in classification performance is observed. It remains unclear if their found spurious features are harmful, that is the feature alone triggers the decision for that class.

Interpretability methods: In recent years several interpretability methods have been proposed e.g., attribution methods such as GradCAM [57], Shapley values [42], Relevance Propagation [7], and LIME [51]. The use of these methods for the detection of spurious features has been analyzed in [2, 1] with mixed success and it has been argued that interpretability methods are not robust [19, 64, 27]. However, attribution methods work better for robust classifiers due to more interpretable gradients [22]. Another technique is counterfactual explanations [76, 75] which are difficult to generate for images due to the similarity to adversarial examples [68]. Thus visual counterfactual explanations are realized via manipulation of a latent space [56] or in image space [53, 6, 13] for an adversarially robust classifier. Visual counterfactuals for non-robust classifiers using diffusion models [35, 65, 33, 18] have been proposed in [5]. ImageNet: ImageNet [52] suffers from several shortcomings: apart from an inherent dataset bias [71], semantically overlapping or even identical class pairs were reported [34, 73, 10], e.g., two classes "maillot", "sunglass" vs "sunglasses", "notebook" vs "laptop" etc. We disregard such trivial cases of dataset contamination and focus on classes with harmful spurious features, in particular ones where only a small portion of the training set is contaminated.

#### **3.** Spurious features

A proper definition of spurious features is difficult. We describe two settings of harmful spurious features which appear in this paper. We denote by  $C_k$  the set of all images containing objects belonging to class k (assuming for simplicity that we have a deterministic problem and ignoring multi-labels). Let S be the set of all images containing a feature s (e.g. a bird feeder). It is a correlated feature for class k when  $C_k \cap S$  and  $S \setminus C_k$  are non-empty, i.e. the feature occurs frequently with the class object but there is no causal implication that appearance of s implies the appearance of the class object (a bird feeder in the image does not imply presence of a hummingbird). A correlated feature becomes a spurious feature when the classifier picks it up as feature of this class. Not every spurious feature is immediately harmful, even humans use context information [25] to get more confident in a decision. However, a spurious feature is *harmful* if the spurious feature alone is enough to trigger the decision for the corresponding class without the class object being present in the image. We consider two scenarios for a harmful spurious feature shown in Fig. 2.



Figure 2: **Type of harmful Spurious Feature:** Left: The spurious feature *s* is taken up by the classifier which predicts class k on  $C_k \cup S$  instead only on  $C_k$ . Right: The spurious feature *s* is shared between classes but appears more often in class k. The classifier associates S with class k and thus predicts class k also on  $C_l \cap S$  instead of class l.

**Spurious Class Extension:** For this type of spurious feature (left in Fig. 2) the classifier picks up the spurious feature s for class k and predicts the class k even on  $S \setminus C_k$  with high confidence (prediction of "hummingbird" for images showing a bird feeder but no hummingbird). The classifier predicts class k beyond its actual domain  $C_k$  and thus we call this a spurious class extension. While this spurious feature does not necessarily hurt in terms of test performance it can easily lead to completely unexpected behavior when the classifier is deployed in the wild.

**Spurious Shared Feature:** Here, two classes  $C_k$  and  $C_l$  share a spurious feature s (e.g., "water jet" for the classes "fireboat" and "fountain"). As there are more training images with feature s in  $C_k$  than in  $C_l$  the classifier associates S with class k and predicts class k for  $S \cap C_l$ .

The two types of harmful spurious features are not exclusive. A shared spurious feature s can at the same time lead to a spurious class extension, e.g., the object bird feeder leads to a spurious class extension of the class "hummingbird" (see Fig. 4) to images of bird feeders without hummingbirds. In the training set the hummingbird" but the hummingbird feeder has parts which mimic flowers and flowers are a shared spurious feature with bees. In Fig. 4 right top row, images of bees on a bird feeder are classified as "hummingbird" instead of "bee", so the spurious feature is strong enough to override the decision for the true class "bee" (spurious shared feature).

# 4. Finding spurious features via neural PCA and associated feature visualizations

First, we define our class-wise neural PCA (NPCA) which allows us to find diverse subpopulations in the train-

ing data, e.g., we checked that the bird feeder for "hummingbird" is visible in 15% of the training images (component 2), while another 15% contain a part of it (component 3), see Fig. 3 or, for more examples, App. F. Then we introduce our neural PCA feature visualization (which requires an adversarially robust model) and how we select NPCA components for human inspection. The identification of spurious features requires minimal human supervision, and our effective setup allows us to screen all ImageNet classes.

Adversarially robust model: Similar to [61], we use an adversarially robust model to find spurious features in ImageNet. The reason for this is that robust models have generative properties [74, 53, 6, 61, 13] in the sense that maximizing the predicted probability of a class in a neighborhood of an image leads to semantically meaningful changes. They also have more informative gradients [22] and thus attribution maps such as GradCAM [57] work better. We use the multiple-norm robust model of [13] as they claim it has the best generative properties. The generative properties of robust models are mainly used for the neural PCA feature visualization where we maximize the NPCA component of a class starting from a gray image. If a spurious feature appears without the class object, this is a strong indicator of a harmful spurious feature. A non-robust model would only produce semantically meaningless adversarial noise, hence we need a robust model for this part of our detection pipeline. Our detected spurious features are not specific to the robust model. We show that SOTA ImageNet models share the same spurious features (Fig. 4 and Sec. 7.2).

**Class-wise neural PCA:** Let  $(x_i, y_i)_{i=1}^N$  be the training set, where  $y_i \in \{1, ..., K\}$  and K is the number of classes. We consider features of the penultimate layer  $\phi(x) \in \mathbb{R}^D$ of a neural network for an input x. For a given class k and its associated weights  $w_k \in \mathbb{R}^D$  in the final layer, we define

$$\psi_k(x) = w_k \odot \phi(x). \tag{1}$$

where  $\odot$  is the componentwise product. Let  $b \in \mathbb{R}^K$  be the bias vector of the final layer then the logit  $f_k$  of class k is:

$$f_k(x) = \sum_{j=1}^{D} w_{kj} \,\phi(x)_j + b_k = \langle \mathbf{1}, \psi_k(x) \rangle + b_k.$$
(2)

Let  $I_k$  be the index set of the training set of class k and  $\bar{\psi}_k$  the class-wise mean,  $\bar{\psi}_k = \frac{1}{|I_k|} \sum_{s \in I_k} \psi_k(x_s)$ . The *class-wise neural PCA* allows us to identify variations in the set  $\{\psi(x_r)\}_{r \in I_k}$  arising due to small subpopulations in the training set. In the class-wise neural PCA, we compute eigenvectors of the class-wise covariance matrix,

$$C = \sum_{s \in I_k} \left( \psi_k(x_s) - \bar{\psi_k} \right) \left( \psi_k(x_s) - \bar{\psi_k} \right)^T.$$
(3)

The eigenvectors,  $v_1, \ldots, v_D$  form an orthonormal basis of  $\mathbb{R}^D$  and we write  $\psi_k(x) - \overline{\psi}_k$  in this basis,

$$\psi_k(x) - \bar{\psi}_k = \sum_{l=1}^D v_l \left\langle \psi_k(x) - \bar{\psi}_k, v_l \right\rangle, \quad (4)$$

Top 5 neural PCA components (ours)



Figure 3: **Top 5 Neural PCA components for class hummingbird:** first row shows our neural PCA feature visualization (NPFV), second row shows four most activating training images of each NPCA component and the last row GradCAM for the NPCA component. Our NPCA components capture different subpopulations in the training data. Comp. 2 is identified as spurious feature "bird feeder", see NPFV and most activating training images (see also Fig. 4).

and define

$$\alpha_l^{(k)}(x) = \langle \mathbf{1}, v_l \rangle \left\langle \psi_k(x) - \bar{\psi}_k, v_l \right\rangle, \tag{5}$$

The logit  $f_k(x)$  of the k-th class can then be written as

$$f_k(x) = \sum_{l=1}^{D} \alpha_l^{(k)}(x) + \left< \mathbf{1}, \bar{\psi}_k \right> + b_k.$$
 (6)

Thus for a given x, we can interpret  $\alpha_l^{(k)}(x)$  as the contribution of the neural PCA component l of class k to the logit  $f_k(x)$  of class k since the term  $\langle \mathbf{1}, \bar{\psi}_k \rangle + b_k$  is constant for all inputs. Based on this we introduce a mitigation technique for spurious features without retraining in Sec. 5.

**Neural PCA Feature Visualization:** To identify semantic features corresponding to our neural PCA component l, we show the training images which attain the maximal values of  $\alpha_l^{(k)}(x)$ . Additionally, we generate an image  $z_l^{(k)}$ , which we call the **Neural PCA Feature Visualization** (**NPFV**) of feature l of class k, by maximizing  $\alpha_l^{(k)}(x)$ :

$$z_l^{(k)} = g + \mathrm{argmax}_{\|\delta\|_2 \leq \epsilon} \ \alpha_l^{(k)}(g+\delta),$$

where g is a gray image (all channels equal to 0.5). Thus we maximize the feature  $\alpha_l^{(k)}$  outgoing from a non-informative and unbiased initialization g. The optimization problem is solved using adaptive projected gradient descent (APGD) [16] with 200 steps. The budget for changes,  $\epsilon = 30$ , is small to avoid the overactivation of feature attacks maximizing the output of individual neurons [21, 63, 61], see

Fig. 8. In Fig. 4 we show for each identified spurious feature, the corresponding NPFV, e.g., for "hummingbird" one can see the bird feeder but no hummingbird. The NPCA components together with the maximally activating training images are in principle sufficient to identify spurious features, but the NPFV is very useful to judge how *harmful* a spurious feature is. Therefore, we use an adversarially robust model, since a non-robust model would yield semantically meaningless adversarial noise as NPFV.

Selection of neural PCA components for human inspection: The penultimate layer of the robust ResNet50 we are using has 2048 neurons. Thus it is infeasible (and unnecessary) to investigate all neural PCA components. A strong criterion that one has found a harmful spurious feature is i) the NPFV shows mainly the spurious feature and not the class, and ii) the NPFV has high confidence. If ii) is not satisfied, then the NPFV is a spurious feature the classifier may have picked up, but it is not harmful in the sense that this feature alone causes the classifier to choose that class. Moreover, we noticed that the eigenvalues of the neural PCA, and the corresponding  $\alpha$  values, decay quickly. Thus we compute the NPFV for the top 128 neural PCA components (having maximal variance) and then select the ten components which realize the highest confidence for their NPFV in the corresponding class. Note that we do not optimize the confidence when generating the NPFV but only  $\alpha_l^{(k)}(x)$  which is part of the logit of the k-th class.

Identification of spurious neural PCA components via human supervision: For each ImageNet class k we show the human labeler the top 10 components. For each component l we show the NPFV  $z_l^{(k)}$  and the 5 training images  $x_r$  of class k with the largest values of  $\alpha_l^{(k)}(x_r)$ . Moreover, we compute GradCAM [57] images for the NPFV and the five training images using the NPCA component  $\alpha_l^{(k)}$ as score. The human marks a component as spurious if i) the NPFV shows dominantly an object not belonging to the class ii) the five training images show consistently this object, iii) the GradCAM activations are primarily not on the class object. The setup shown to the human labeler can be seen in App. B. The labeling of one class takes on average about 45 seconds, so the full labeling of all ImageNet classes took about 13 hours. The human labeler (researcher in machine learning) found in total 337 spurious components. Another human labeler checked all of them and removed spurious features in case of disagreement, resulting in 322 spurious features from 233 ImageNet classes.

#### 5. SpuFix - Mitigation of spurious features

Once the spurious features are identified, the question is how one can mitigate that the classifier relies on them. One way is to identify the training images containing the spurious feature and then discard or downweight them during



Figure 4: **Spurious features (ImageNet):** found by human labeling of our neural PCA components. For each class we show 5 random train. images (top left), the neural PCA Feature Visual. (NPFV) and 4 most activating train. images for the spurious feature component (bottom left). Right: four ImageNet models classify images **showing only the spurious feature but no class object** as this class.

training. However, this would require relabeling all spurious ImageNet classes which is not feasible. We could order the training set according to the value  $\alpha_l^{(k)}$  of the corresponding neural PCA component which indicates how much of the spurious feature an image contains. While this would speed up the process significantly, it would still require a significant amount of manual relabeling. Can one do it also without any additional labeling? Yes, as described in Sec. 4 we can rewrite the logit of the k-th class as

$$f_k(x) = \sum_{l=1}^{D} \alpha_l^{(k)}(x) + \left\langle \mathbf{1}, \bar{\psi}_k \right\rangle + b_k.$$
(7)

For a spurious component l of class k, we use  $\min{\{\alpha_l^{(k)}, 0\}}$  instead of  $\alpha_l^{(k)}$  to remove its positive contribution from the logit (negative contributions are semantically different). After removal of the spurious features, the new logit becomes

$$f_{k}^{SpuFix}(x) = f_{k}(x) - \sum_{l \in \mathcal{S}_{k}} \max\{\alpha_{l}^{(k)}(x), 0\}$$
(8)

where  $S_k$  is the set of spurious NPCA components of class k. We denote this method as **SpuFix**. It significantly reduces dependence on spurious features, see Sec. 7.3.

**Transfer of SpuFix to any ImageNet classifer:** As described in Sec. 3, harmful spurious features are a result of subpopulations in the training data. While not every spurious correlation will be picked up by every model, most of our detected spurious features generalize to a wide range of classifiers (see Sec. 7). In the following, we show how Spu-Fix can be transferred to any given ImageNet classifier  $\tilde{f}$  for which  $\tilde{f}_k$  denotes the logit and  $\tilde{\psi}_k$  the weighted penultimate layer of class k. The goal is to find a direction b in the weighted feature space  $\tilde{\psi}_k$  of  $\tilde{f}$  for every spurious NPCA component l corresponding to the eigenvector  $v_l$  of the original model, resp.  $\alpha_l^{(k)}(x)$ , and then truncate its positive component. To find this direction we maximize the covariance of the projection onto b and  $\alpha_l^{(k)}$  over the training images of class k:

$$b_l^{(k)} = \underset{\|b\|_2=1}{\operatorname{argmax}} \sum_{s \in I_k} \left\langle b, \tilde{\psi}_k(x_s) - \bar{\tilde{\psi}}_k \right\rangle \alpha_l^{(k)}(x_s).$$
(9)

which has a closed form solution

$$b_{l}^{(k)} = \frac{\sum_{s \in I_{k}} \left( \tilde{\psi}_{k}(x_{s}) - \tilde{\psi}_{k} \right) \alpha_{l}^{(k)}(x_{s})}{\left\| \sum_{s \in I_{k}} \left( \tilde{\psi}_{k}(x_{s}) - \bar{\tilde{\psi}}_{k} \right) \alpha_{l}^{(k)}(x_{s}) \right\|_{2}}.$$
 (10)

In contrast to the eigenvectors  $v_l$ , the matched vectors  $b_l^{(k)}$  are not necessarily orthogonal. Thus before truncation the centered features  $\tilde{\psi}_k(x) - \tilde{\psi}_k$  need to be projected onto the subspace spanned by the  $b_l^{(k)}$  and represented in the non-orthogonal basis  $\{b_l^{(k)}\}_{l \in S_k}$ . We denote this representation

by  $P^{(k)}(x)$  (details in C.1). The logit of class k of the Spu-Fix version of  $\tilde{f}$  is then:

$$\tilde{f}_k^{SpuFix}(x) = \tilde{f}_k(x) - \sum_{l \in \mathcal{S}_k} \max\left\{ \left\langle \mathbf{1}, b_l^{(k)} \right\rangle P_l^{(k)}(x), 0 \right\}.$$
(11)

In the case where  $\tilde{f} = f$  (the robust model), we recover the original SpuFix truncation in (8) (see C.2). It turns out that SpuFix is even effective when the architecture is quite different from the ResNet50 we used for detection, e.g. ViT or VOLO, see Sec. 7.3 and Table 1 and 2.

# 6. Comparison to neural features of [61]

We compare our NPCA framework to the method of [61] to detect spurious features for ImageNet. As model, they use a  $\ell_2$ -robust ResNet50. Let  $J_k$  be the set of training images classified as class k. [61] define the *j*-th component  $m_i^{(k)}$  of the class-wise mean over predictions,

$$m^{(k)} = \frac{1}{|J_k|} \sum_{x_s \in J_k} \psi_k(x_s),$$
 (12)

as the importance of the *j*-th neuron for class k.<sup>1</sup> Then, they order the neurons of the penultimate layer according to the score  $m_i^{(k)}$  and consider the top-5 neurons of each class. The main difference to our approach is that they assume single neurons with maximal influence on the mean are capturing spurious features whereas our NPCA components are linear combinations of neurons that capture the variance around the mean. Given that the ResNet50 has only 2048 neurons for 1000 classes, some neurons are labeled as core and spurious feature for multiple classes simultaneously, even though the images are quite different. A major advantage of NPCA is that due to the orthogonality of the PCA components, we identify diverse subpopulations in the training data. As [61] use no constraints for the neurons, they often find very similar subpopulations. Hence, one may miss spurious subpopulations when only checking the top-5 components, see Fig. 5. Another difference is that they maximize the score  $m_i^{(k)}$  for the training images  $x_r$ 

$$w_j^{(k)} = x_r + \operatorname{argmax}_{\|\delta\|_2 \le \epsilon} m_j^{(k)}(x_r + \delta),$$

whereas we maximize our NPCA component  $\alpha_l^{(k)}(x)$  starting from a gray image to introduce no bias. Thus, we check if the classifier produces the spurious feature and not only enhances it on an image showing it already.

The third difference is that [61] want to identify *any* spurious feature while our goal is to find *harmful* ones. Thus, they use weaker criteria for deciding if a neuron shows a spurious feature: main criterion is if the neural activation

<sup>&</sup>lt;sup>1</sup>The top-5 neurons do not change when using  $I_k$  instead of  $J_k$ .



Figure 5: Comparison of top-5 NPCA and top-5 neurons for class hummingbird for the robust model of [61]: Our NPCA components identify diverse subpopulations in the training set whereas the top neurons show similar ones and the spurious bird feeder is not detected, see also App. A.

map based on  $m_j^{(k)}$  is off the class object according to a majority vote of 5 human labelers. The visualizations  $w_j^{(k)}$ are only used if the activation maps are inconclusive. In contrast, we require i) our NFPV to show the spurious feature, ii) the GradCAM based on  $\alpha_l^{(k)}$  highlights mainly the spurious feature, and iii) our human labelers have to *agree* that the NPCA component is a harmful spurious feature.

As our criteria are more strict (in particular, that the NFPV shows the spurious feature is a strong criterion), it is not surprising that [61] find more spurious features (630) in 357 classes) than we do (322 in 233 classes). Moreover, the employed models are different and we examine top-10 NPCA components whereas they check top-5 neurons. Thus, the comparison is difficult and our novel dataset "SpuriousImageNet" could be biased towards spurious features which only we found. Hence, we compute our top-5 NPCA components for their robust ResNet50 for a direct comparison to their top-5 neurons and their found spurious features. In Fig. 5 we compare them for the class hummingbird where they do not find the spurious feature "bird feeder". In general, we observe that our found subpopulations are more diverse and thus we find more spurious features than they do when using their weaker criteria. In App. E we do an extensive comparison for all classes.

## 7. Experiments

In this section, we provide a qualitative and quantitative evaluation of our 322 detected spurious features in ImageNet, see Sec. 4. For the quantitative evaluation, we create the dataset "Spurious ImageNet", which allows checking the reliance of a given ImageNet classifier on spurious features. We also evaluate our mitigation strategy "SpuFix" which does not require additional labels or retraining of the classifier and can be transferred to other image classifiers.

#### 7.1. Qualitative evaluation

For the qualitative evaluation, we visualize some of our found 322 spurious features, see Fig. 4. For each class, we show five random training images, the NPFV, and the four most activating training images of the neural PCA component labeled to be spurious. Additionally, we always show ten images which only show the spurious feature but not the actual class e.g., only the bird feeder (spurious) but no hummingbird (class). All ten images are classified as the corresponding class for the robust classifier we have used to compute the NPCA components and three non-robust ImageNet classifiers (ResNext101, Eff.Net B5, ConvNext-B, see also Tab. 2). This shows that our spurious features generalize from the robust classifier to SOTA ImageNet classifiers, indicating that the found spurious features are mainly due to the design of the training set, rather than failures in model training. Our novel validation by collecting real images with the spurious feature but without the class object which are consistently classified as this class directly shows the impact of harmful spurious features and has the advantage that it does not introduce artifacts via masking nor requires expensive pixel-wise segmentations.

Image Collection: The images showing only the spurious feature were obtained by sorting the 9 million images of OpenImages [38] by the value  $\alpha_l^{(k)}(x)$  of the neural PCA component. We check the top 625 retrieved images classified by the robust classifier as the corresponding class if they are all classified as the same class by the additional non-robust classifiers and do not show the corresponding class. This is a quite strict criterion as spurious features can be shared across classes, e.g., twigs for birds, and thus agreement of classifiers is not granted and the images can show the spurious feature and the true class. Nevertheless, this procedure yields between 77 ("hummingbird") and 179 ("freight car") images of which we show a selection. For "hummingbird", a lot of these images show red flowers (without a hummingbird) which makes sense as the NPFV displays features of red flowers and due to the bias that OpenImages does not contain many images of hummingbird feeders. In these cases, we additionally retrieve Flickr images with appropriate text queries e.g., "hummingbird feeder" and filter them.

**Spurious Class Extension:** For "hummingbird", "freight car", and "koala" the spurious features significantly extend the predictions beyond the actual class (see Fig. 2). Bird feeders are classified as hummingbirds, graf-



Figure 6: **Spurious Score:** we plot the AUC of different models for 7 out of 100 classes in "Spurious ImageNet" (see Fig 13-15): hummingbird (bird feeder/red flowers), freight car (graffiti), koala (plants/trees), fireboat (water jet), flagpole (US flag), gondola (house/river), and bookshop (storefront). The spurious features for hummingbird, freight car and koala which were not detected in [61, 62] are also spurious for their robust ResNet50 [61]. Training on ImageNet 21k or Fine-tuning from 21k (1kFT21k) decreases dependence on harmful spurious features but classes like flagpole and bookshop remain strongly affected. Our cheap mitigation strategy SpuFix improves the AUCs significantly for both robust ResNet50 but also improves the ViT-B variants trained/fine-tuned on ImageNet1k, especially for the difficult classes flagpole and bookshop.



Figure 7: **Spurious ImageNet:** sample images from the dataset for 6 out of 100 classes showing the spurious feature but not the class object, see also App. **G**.

fiti as freight cars, and (eucalyptus) plants as koalas. This class extension cannot be detected by monitoring test performance and thus is likely to be noticed only after deployment. For "hummingbird", we see in Fig. 4 two images with bees on the bird feeder where "bee" is an ImageNet class (also a grasshopper for "koala"). Nevertheless, the spurious "bird feeder" feature of "hummingbird" overrules "bee" even though no hummingbird is present.

**Spurious Shared Feature:** The spurious feature "water jet" is shared among the classes "fireboat" and "fountain". It appears more frequently for "fireboat" (see Fig. 2) which leads to an in-distribution shift where now a large number of images of the "fountain"-class with a water jet are wrongly classified as "fireboat". The spurious feature "water jet" for "fireboat" has been found also in the Salient ImageNet dataset [61, 62]. However, they did not find spurious features for freight car and koala (in App. A we do a comparison). More examples are in App. F.

#### 7.2. The Spurious ImageNet dataset

A key contribution of this paper is our novel evaluation of spurious features for image classifiers without requiring pixel-wise annotations [48, 59] or having to rely on the validity of neural heatmaps [61]. Instead, we use images from OpenImages to show that images only containing the spurious feature but not the class object are classified as this class. This has the advantage that we consider real images and thus provide a realistic impression of the performance of ImageNet classifiers in the wild. Adding noise [61] or masking [48, 44] image regions requires pixel-wise accurate annotations which are labor-expensive, masking only the object still contains shape information, and using masks avoiding this, e.g., a bounding box around the object, can hide a significant portion of the image which is unrealistic.

To allow for a quantitative analysis of the influence of spurious features on ImageNet classifiers, we collected images similar to the ones shown to illustrate the spurious features in Fig. 4. The images are chosen such that they show the spurious feature but not the class object. The only difference is that we relax the classification condition and only require two of the four classifiers (robust ResNet50, ResNext101, EfficientNet-B5, ConvNext-B) to predict the corresponding class. We select 100 of our spurious features and for each collect 75 images from the top-ranked images in OpenImages according to the value of  $\alpha_1^{(k)}$  for which two human labelers agree that they contain the spurious feature but not class k and two out of four classifiers predict class k. We call the dataset **Spurious ImageNet** as it allows to check the dependence on spurious features with real images for ImageNet classifiers, see Fig. 7 and App. G for samples.

**Spurious Score:** A classifier f not relying on the spurious feature should predict a low probability for class k for the Spurious ImageNet samples, especially compared to ImageNet test set images of class k. Thus, for each class, we measure the AUC (area under the curve) for the separation

of images with the spurious features but not showing class k versus test set images of class k according to the predicted probability for class k. A classifier not depending on the spurious feature should attain a perfect AUC of 1, whereas a value significantly below 1 shows strong reliance. We report the mean AUC (mAUC) over all 100 classes in Tab. 1. All ImageNet models trained only on ImageNet1k are heavily influenced by spurious features. Thus, spurious features are mainly a problem of the training set rather than the classifier, and spurious features found with an adversarially robust model transfer to other ImageNet classifiers.

Pre-training on larger datasets: Some spurious features such as flag (flag pole), bird feeder (hummingbird), and eucalyptus (koala) are classes in ImageNet21k. Therefore, they should no longer be spurious for the other classes. Thus, we test if ImageNet1k-classifiers fine-tuned from an ImageNet21k model are less reliant on spurious features. The results in Tab. 1 and Fig. 6 suggest that the influence of spurious features is damped but they are far from being free of them. To check how much is lost due to fine-tuning we evaluate a ViT-B trained on ImageNet21k which has a mean AUC of 0.931 whereas the fine-tuned model has 0.917. This shows that fine-tuning does not hurt much. While finetuning from ImageNet21k improves the mean AUC, for several classes the dependence on spurious features is still significant, see also Fig. 16 how one has to be careful in the interpretation of higher AUC values. In addition to ImageNet21k, we also evaluate models trained on other large image datasets (JFT-300M[28], YFFC-100M, 1B Instagram[84], MIM[24], LAION-2B and LAION-400M[54]) using self-supervised learning or which are based on CLIP [49]. However, these models also do not achieve better spurious scores (Tab. 1 and Tab. 2). We evaluate a large number of SOTA models in App. D.

#### 7.3. Evaluation of mitigation technique SpuFix

Fixing spurious features is a non-trivial task and can require a substantial labeling effort. We evaluate our Spu-Fix from Sec. 5 that does not require retraining or additional labels. The positive effect of this fix of spurious features (SpuFix) can be seen in Tab. 1 and Fig. 6. Compared to the original robust ResNet50 with a spurious mAUC of 0.630, the SpuFix version has a significantly better spurious mAUC of 0.763. Test set accuracy reduces by 0.6% but this is a rather positive effect, as several of the additional errors arise since the robust ResNet50 uses spurious features for its decision, e.g., for classes like "balance beam" or "puck" the class object is often not visible in the cropped test set images. In Table 2 we provide a large scale evaluation of the transfer of SpuFix to SOTA ImageNet models. We observe a consistently better mAUC on Spurious ImageNet, even for very large models fine-tuned from 21k or trained on other large datasets, e.g. SpuFix improves the mAUC of VOLO-

	Or	iginal	SpuFix				
	INet	SpurIN	INet	SpurIN			
Model	Acc. $\uparrow$	mAUC $\uparrow$	Acc. $\uparrow$	mAUC $\uparrow$			
ImageNet1k							
Rob. ResNet50	57.4%	0.630	56.8%	0.763			
Rob. ResNet50[61]	57.9%	0.651	57.2%	0.764			
ConvNeXt-L[41]	84.8%	0.803	84.8%	0.819			
ViT-B AugReg[66]	81.1%	0.850	81.1%	0.859			
VOLO-D5 512[85]	87.1%	0.882	87.1%	0.907			
ImageNet21kFT1k							
EfficientNetv2-L[70]	86.8%	0.893	86.8%	0.898			
ConvNeXt-L[41]	87.0%	0.910	87.0%	0.913			
ViT-B AugReg[66]	86.0%	0.917	85.9%	0.925			
BEiT-L\16[8]	88.6%	0.921	88.6%	0.927			
LAION-2B							
CNeXt-L CLIP 384[49]	87.8%	0.879	87.9%	0.884			
ViT-L\14 CLIP[49]	88.2%	0.912	88.2%	0.914			
MIM							
EVA-G\14 CLIP 560[24]	89.8%	0.919	89.8%	0.925			
ImageNet21k							
ConvNeXt-L[41]	-	0.943	-	0.943			
ViT-B AugReg[66]	-	0.931	-	0.931			

Table 1: Quantitative Evaluation on Spurious ImageNet: ImageNet classifiers of different training modalities depend on spurious features in varying strength. The mAUC is the mean of AUCs for the separation of images containing the spurious feature but not class k versus test images of class k with the predicted probability of class k as score.

D5 (87.1% acc.) trained only on 1k by 2.5%, or by 0.6% for EVA-G\14 CLIP 560 trained on MIM (91.9% acc.), as well as BeiT-L\16 fine-tuned from 21k (88.6% acc.) by 0.6%. Thus even SOTA models profit from our SpuFix with negligible difference in accuracy ( $\leq 0.1\%$ ) and thus the use of SpuFix is recommended for any ImageNet model.

#### 8. Conclusion

We have shown that large-scale identification of spurious features is feasible with our neural PCA components and neural PCA feature visualizations. With "Spurious ImageNet" we introduced a novel dataset to evaluate the dependence of ImageNet classifiers on spurious features based on real images. We demonstrated that our SpuFix method mitigates the dependence on harmful spurious features for any ImageNet classifier without costly labeling or re-training.

Acknowledgements We acknowledge support by the DFG, Project number 390727645, the Carl Zeiss Foundation, project "Certification and Foundations of Safe Machine Learning Systems in Healthcare". The authors thank the IMPRS-IS for supporting YN.

## References

- [1] Julius Adebayo, Michael Muelly, Hal Abelson, and Been Kim. Post hoc explanations may be ineffective for detecting unknown spurious correlation. In *ICLR*, 2022.
- [2] Julius Adebayo, Michael Muelly, Ilaria Liccardi, and Been Kim. Debugging tests for model explanations. In *NeurIPS*, 2020.
- [3] D. Amodei, C. Olah, J. Steinhardt, P. Christiano, J. Schulman, and D. Mane. Concrete problems in AI safety. arXiv:1606.06565v2, 2016.
- [4] Christopher J. Anders, Leander Weber, David Neumann, Wojciech Samek, Klaus-Robert Müller, and Sebastian Lapuschkin. Finding and removing clever hans: Using explanation methods to debug and improve deep models. *Information Fusion*, 77:261–295, 2022.
- [5] Maximilian Augustin, Valentyn Boreiko, Francesco Croce, and Matthias Hein. Diffusion visual counterfactual explanations. In *NeurIPS*, 2022.
- [6] Maximilian Augustin, Alexander Meinke, and Matthias Hein. Adversarial robustness on in- and out-distribution improves explainability. In ECCV, 2020.
- [7] David Baehrens, Timon Schroeter, Stefan Harmeling, Motoaki Kawanabe, Katja Hansen, and Klaus-Robert Müller. How to explain individual classification decisions. *JMLR*, 2010.
- [8] Hangbo Bao, Li Dong, and Furu Wei. Beit: Bert pre-training of image transformers. In *ICLR*, 2022.
- [9] Sara Beery, Grant van Horn, and Pietro Perona. Recognition in terra incognita. In *ECCV*, 2018.
- [10] Lucas Beyer, Olivier J Hénaff, Alexander Kolesnikov, Xiaohua Zhai, and Aäron van den Oord. Are we done with ImageNet? arXiv:2006.07159, 2020.
- [11] B. Biggio, I. Corona, D. Maiorca, B. Nelson, N. Srndic, P. Laskov, G. Giacinto, and F. Roli. Evasion attacks against machine learning at test time. In *ECML/PKKD*, 2013.
- [12] Alceu Bissoto, Eduardo Valle, and Sandra Avila. Debiasing skin lesion datasets and models? not so fast. arXiv:2004.11457, 2020.
- [13] Valentyn Boreiko, Maximilian Augustin, Francesco Croce, Philipp Berens, and Matthias Hein. Sparse visual counterfactual explanations in image space. In *GCPR*, 2022.
- [14] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In *NeurIPS*, 2020.
- [15] Myung Jin Choi, Antonio Torralba, and Alan S. Willsky. Context models and out-of-context objects. *Pattern Recognition Letters*, 33(7):853–862, 2012.
- [16] Francesco Croce and Matthias Hein. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *ICML*, 2020.

- [17] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *CVPR workshop*, pages 702– 703, 2020.
- [18] Prafulla Dhariwal and Alex Nichol. Diffusion models beat gans on image synthesis. In *NeurIPS*, 2021.
- [19] Ann-Kathrin Dombrowski, Maximillian Alber, Christopher Anders, Marcel Ackermann, Klaus-Robert Müller, and Pan Kessel. Explanations can be manipulated and geometry is to blame. In *NeurIPS*, 2019.
- [20] 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. In *ICLR*, 2021.
- [21] Logan Engstrom, Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Brandon Tran, and Aleksander Madry. Adversarial robustness as a prior for learned representations, 2019.
- [22] Christian Etmann, Sebastian Lunz, Peter Maass, and Carola-Bibiane Schönlieb. On the connection between adversarial robustness and saliency map interpretability. In *ICML*, 2019.
- [23] Sabri Eyuboglu, Maya Varma, Khaled Saab, Jean-Benoit Delbrouck, Christopher Lee-Messer, Jared Dunnmon, James Zou, and Christopher Ré. Domino: Discovering systematic errors with cross-modal embeddings. In *ICLR*, 2022.
- [24] Yuxin Fang, Wen Wang, Binhui Xie, Quan Sun, Ledell Wu, Xinggang Wang, Tiejun Huang, Xinlong Wang, and Yue Cao. Eva: Exploring the limits of masked visual representation learning at scale. arXiv preprint arXiv:2211.07636, 2022.
- [25] Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A. Wichmann. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673, 2020.
- [26] R. Geirhos, P. Rubisch, C.Michaelis, M. Bethge, F. A. Wichmann, and W. Brendel. Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness. In *ICLR*, 2019.
- [27] Amirata Ghorbani, Abubakar Abid, and James Zou. Interpretation of neural networks is fragile. In *AAAI*, 2019.
- [28] A Gupta, C Sun, A Shrivastava, and S Singh. Revisiting the unreasonable effectiveness of data. *arXiv preprint arXiv:1707.02968*, 2017.
- [29] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016.
- [30] M. Hein, M. Andriushchenko, and J. Bitterwolf. Why ReLU networks yield high-confidence predictions far away from the training data and how to mitigate the problem. In *CVPR*, 2019.
- [31] D. Hendrycks and T. Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. In *ICLR*, 2019.
- [32] D. Hendrycks and K. Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. In *ICLR*, 2017.
- [33] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. In *NeurIPS Workshop*, 2021.

- [34] Sara Hooker, Yann Dauphin, Aaron Courville, and Andrea Frome. Selective brain damage: Measuring the disparate impact of model pruning. In *ICLR*, 2020.
- [35] Pieter Abbeel Jonathan Ho, Ajay Jain. Denoising diffusion probabilistic models. In *NeurIPS*, 2020.
- [36] Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Joan Puigcerver, Jessica Yung, Sylvain Gelly, and Neil Houlsby. Big transfer (bit): General visual representation learning. In ECCV, 2020.
- [37] A. Kurakin, I. J. Goodfellow, and S. Bengio. Adversarial examples in the physical world. In *ICLR Workshop*, 2017.
- [38] Alina Kuznetsova, Hassan Rom, Neil Alldrin, Jasper Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Malloci, Alexander Kolesnikov, Tom Duerig, and Vittorio Ferrari. The open images dataset v4: Unified image classification, object detection, and visual relationship detection at scale. *IJCV*, 2020.
- [39] Ze Liu, Han Hu, Yutong Lin, Zhuliang Yao, Zhenda Xie, Yixuan Wei, Jia Ning, Yue Cao, Zheng Zhang, Li Dong, et al. Swin transformer v2: Scaling up capacity and resolution. In *CVPR*, 2022.
- [40] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, 2021.
- [41] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *CVPR*, 2022.
- [42] Scott M. Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *NeurIPS*, 2017.
- [43] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *ICLR*, 2018.
- [44] Mazda Moayeri, Sahil Singla, and Soheil Feizi. Hard ImageNet: Segmentations for objects with strong spurious cues. In *NeurIPS Datasets and Benchmarks Track*, 2022.
- [45] A. Nguyen, J. Yosinski, and J. Clune. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *CVPR*, 2015.
- [46] Zhiliang Peng, Li Dong, Hangbo Bao, Qixiang Ye, and Furu Wei. Beit v2: Masked image modeling with vector-quantized visual tokenizers. arXiv preprint arXiv:2208.06366, 2022.
- [47] Ed Pizzi, Sreya Dutta Roy, Sugosh Nagavara Ravindra, Priya Goyal, and Matthijs Douze. A self-supervised descriptor for image copy detection. *Proc. CVPR*, 2022.
- [48] Gregory Plumb, Marco Tulio Ribeiro, and Ameet Talwalkar. Finding and fixing spurious patterns with explanations. *Transactions on Machine Learning Research (TMLR)*, 2022.
- [49] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *ICML*, 2021.
- [50] Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do ImageNet classifiers generalize to ImageNet? In *ICML*, 2019.

- [51] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "why should i trust you?": Explaining the predictions of any classifier. In *KDD*, 2016.
- [52] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge. *IJCV*, 2015. License: No license specified.
- [53] Shibani Santurkar, Dimitris Tsipras, Brandon Tran, Andrew Ilyas, Logan Engstrom, and Aleksander Madry. Image synthesis with a single (robust) classifier. In *NeurIPS*, 2019.
- [54] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, et al. Laion-5b: An open large-scale dataset for training next generation image-text models. In *NeurIPS*, 2022.
- [55] Christoph Schuhmann, Richard Vencu, Romain Beaumont, Robert Kaczmarczyk, Clayton Mullis, Aarush Katta, Theo Coombes, Jenia Jitsev, and Aran Komatsuzaki. Laion-400m: Open dataset of clip-filtered 400 million image-text pairs. arXiv preprint arXiv:2111.02114, 2021.
- [56] Kathryn Schutte, Olivier Moindrot, Paul Hérent, Jean-Baptiste Schiratti, and Simon Jégou. Using stylegan for visual interpretability of deep learning models on medical images. In *NeurIPS Workshop*, 2020.
- [57] Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *ICCV*, 2017.
- [58] Harshay Shah, Kaustav Tamuly, Aditi Raghunathan, Prateek Jain, and Praneeth Netrapalli. The pitfalls of simplicity bias in neural networks. In *NeurIPS*, 2020.
- [59] Rakshith Shetty, Bernt Schiele, and Mario Fritz. Not using the car to see the sidewalk–quantifying and controlling the effects of context in classification and segmentation. In *CVPR*, 2019.
- [60] Krishna Kumar Singh, Dhruv Mahajan, Kristen Grauman, Yong Jae Lee, Matt Feiszli, and Deepti Ghadiyaram. Don't judge an object by its context: learning to overcome contextual bias. In CVPR, 2020.
- [61] Sahil Singla and Soheil Feizi. Salient imagenet: How to discover spurious features in deep learning? In *ICLR*, 2022.
- [62] Sahil Singla, Mazda Moayeri, and Soheil Feizi. Core risk minimization using salient imagenet. arXiv:2203.15566, 2022.
- [63] Sahil Singla, Besmira Nushi, Shital Shah, Ece Kamar, and Eric Horvitz. Understanding failures of deep networks via robust feature extraction. In *CVPR*, 2021.
- [64] Dylan Slack, Sophie Hilgard, Emily Jia, Sameer Singh, and Himabindu Lakkaraju. Fooling lime and shap: Adversarial attacks on post hoc explanation methods. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, pages 180–186, 2020.
- [65] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *ICLR*, 2021.

- [66] Andreas Steiner, Alexander Kolesnikov, Xiaohua Zhai, Ross Wightman, Jakob Uszkoreit, and Lucas Beyer. How to train your vit? data, augmentation, and regularization in vision transformers. In *TMLR*, 2022.
- [67] Pierre Stock and Moustapha Cisse. Convnets and ImageNet beyond accuracy: Understanding mistakes and uncovering biases. In ECCV, 2018.
- [68] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *ICLR*, 2014.
- [69] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *ICML*, 2019.
- [70] Mingxing Tan and Quoc Le. Efficientnetv2: Smaller models and faster training. In *ICML*, 2021.
- [71] Antonio Torralba and Alexei A Efros. Unbiased look at dataset bias. In CVPR, pages 1521–1528, 2011.
- [72] Hugo Touvron, Matthieu Cord, and Hervé Jégou. Deit iii: Revenge of the vit. In ECCV, 2022.
- [73] Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Andrew Ilyas, and Aleksander Madry. From ImageNet to image classification: Contextualizing progress on benchmarks. In *ICML*, 2020.
- [74] Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. Robustness may be at odds with accuracy. In *ICLR*, 2019.
- [75] Sahil Verma, John P. Dickerson, and Keegan Hines. Counterfactual explanations for machine learning: A review. *arXiv*:2010.10596, 2020.
- [76] Sandra Wachter, Brent Mittelstadt, and Chris Russell. Counterfactual explanations without opening the black box: Automated decisions and the GDPR. *Harvard Journal of Law* & *Technology*, 2018.
- [77] Ross Wightman. Pytorch image models. https://github.com/ rwightman/pytorch-image-models, 2019.
- [78] Eric Wong, Shibani Santurkar, and Aleksander Madry. Leveraging sparse linear layers for debuggable deep networks. In *ICML*, 2021.
- [79] Sanghyun Woo, Shoubhik Debnath, Ronghang Hu, Xinlei Chen, Zhuang Liu, In So Kweon, and Saining Xie. Convnext v2: Co-designing and scaling convnets with masked autoencoders. arXiv preprint arXiv:2301.00808, 2023.
- [80] Kai Yuanqing Xiao, Logan Engstrom, Andrew Ilyas, and Aleksander Madry. Noise or signal: The role of image backgrounds in object recognition. In *ICLR*, 2021.
- [81] Qizhe Xie, Minh-Thang Luong, Eduard Hovy, and Quoc V. Le. Self-training with noisy student improves imagenet classification. In *CVPR*, 2020.
- [82] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *CVPR*, 2017.
- [83] Cong Xu and Min Yang. Adversarial momentum-contrastive pre-training. arXiv preprint, arXiv:2012.13154v2, 2020.
- [84] I Zeki Yalniz, Hervé Jégou, Kan Chen, Manohar Paluri, and Dhruv Mahajan. Billion-scale semi-supervised learning for image classification. arXiv preprint arXiv:1905.00546, 2019.

- [85] Li Yuan, Qibin Hou, Zihang Jiang, Jiashi Feng, and Shuicheng Yan. Volo: Vision outlooker for visual recognition. *PAMI*, 2022.
- [86] John R. Zech, Marcus A. Badgeley, Manway Liu, Anthony B. Costa, Joseph J. Titano, and Eric Karl Oermann. Variable generalization performance of a deep learning model to detect pneumonia in chest radiographs: A cross-sectional study. *PLOS Medicine*, 15(11):1–17, 2018.
- [87] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018.
- [88] Zhuotun Zhu, Lingxi Xie, and Alan L. Yuille. Object recognition with and without objects. In *IJCAI*, 2017.

### **Overview of Appendix**

In the following, we provide a brief overview of the additional experiments reported in the Appendix.

- In App. A, we compare our top-5 Neural PCA components for our robust model vs. the neural features of the Top-5 neurons of [61] for their robust model for classes "Koala", "Indigo-Bunting", and "Mountain Bike", for which [61] do not find spurious features and we do (see App. E for a direct comparison of NPCA for their robust model to their found components as done in Fig. 5)
- In App. B, we explain our labeling setup to create the dataset "Spurious ImageNet" in more detail.
- In App. C.1 and App. C.2, we present the details of transferring SpuFix to other models. In particular, we define the orthogonal projection  $P^{(k)}$  onto the subspace spanned by non-orthogonal vectors and show that the transfer recovers the original SpuFix method when applied to the original model. We validate that the SpuFix improvement is independent of the image collection procedure in Fig. 12.
- In App. D, we use our "Spurious ImageNet" dataset to quantitatively analyze the dependence of the classifiers on spurious components. By doing so, we show that pre-training on larger datasets like ImageNet21k helps to reduce this dependence. We also discuss the empirical results of the SpuFix method.
- In App. E we continue the comparison to [61] from Section 6. As in Fig. 5. we do a direct comparison to their found top-5 neurons by computing the top-5 NPCA components for their robust model. We observe that their top-5 neurons are less diverse than our top-5 NPCA components, see Fig. 17 and Fig. 18.
- In App. F, we extend our qualitative evaluation of the spurious components from Figure 4.
- In App. G, we show random samples from all 100 spurious features in our "Spurious ImageNet" dataset.
- In App. H, we show how we change the predicted class for an image by introducing only spurious features of the target class. To do this automatically, we adapt the Diffusion Visual Counterfactual Explanations (DVCEs) of [5].

# **A. Neural PCA Components**

We illustrate in Fig. 8 that our neural PCA components capture the different subpopulations in the training set better

compared to the neural features of [61]. We find three spurious features: eucalyptus/plants for the class koala, twigs for the class indigo bunting, and forest for mountain bike, which were not found by [61]. Please see the caption of Fig. 8 for more details. Note that for this comparison, we consider the NPCA components computed on our robust ResNet50 which differs from the one used in [61]. See Fig. 19,20 and App. E for a comparison using the same model.

For 46 out of 100 classes in our "Spurious ImageNet" dataset, no spurious feature is reported in [61]. The 46 classes are: tench, indigo bunting, American alligator, black grouse, ptarmigan, ruffed grouse, s.-c. cockatoo, hummingbird, koala, leopard, walking stick, gar, bakery, barbershop, barn, bathtub, beer bottle, bikini, bulletproof vest, bullet train, chain mail, cradle, dam, dumbbell, fountain pen, freight car, hair spray, hamper, hard disc, mountain bike, neck brace, nipple, obelisk, ocarina, pencil box, pill bottle, plastic bag, plunger, pole, pop bottle, quill, radio telescope, shoe shop, shovel, steel drum, cheeseburger. However, note that even if for the same class their and our method report a spurious feature, this need not be the same.

#### **B.** Labeling Setup for Spurious features

In our paper, we have two labeling tasks for two objectives: i) identifying spurious components, and ii) creating "Spurious Imagenet".

**Identifying spurious components.** Fig. 9 illustrates the information shown to the human labeler to identify neural PCA components corresponding to spurious features. This includes the NPFV, the 5 most activating training images, and GradCAM heatmaps, as well as the corresponding class probabilities and  $\alpha_i^{(k)}$  values. The decision, of whether a neural PCA component corresponds to a spurious feature has been made only based on the visualization as shown in Fig. 9.

**Creating "Spurious Imagenet".** To create our "Spurious ImageNet" dataset,

- we selected 100 (spurious component l, class k) pairs, such that for each class we have only one selected spurious component, and sorted all images from OpenImages for which at least two of our four classifiers predict class k according to the value α<sub>l</sub><sup>(k)</sup> of the respective neural PCA component l;
- we have used the open-source tool<sup>2</sup> for labeling images and created three labels "in" (the images that contain the class features), "out" (the images that contain only the spurious and no class features), "trash" (images that are too far from the distribution of the spurious

<sup>&</sup>lt;sup>2</sup>https://github.com/robertbrada/PyQt-image-annotation-tool

#### Koala



Figure 8: Neural PCA feature components vs neural features of [61] for different classes (computed on our own multiple-norm robust model vs. their  $l_2$ -robust model): First row: NPCA feature visualization (NPFV) of our top-5 NPCA components (left), and the feature attacks of the top-5 neurons of [61] (right). Second row: four most activating training images of the components/neurons. Last row: GradCAM for the NPCA components (left) and the neural activation map of [61] (right). For these three classes [61] report no spurious feature. As in Figure 3 our NPCA components are capturing different subpopulations in the training data. Our NPCA Component 3 of Koala shows prominent leaves in the NPFV and neural PCA GradCAM heatmaps and is identified as spurious, similar for our component 3 of Indigo Bunting showing twigs in the NPFV and in the heatmaps, and component 1 for mountain bike where the forest appears in the NPFV and is active in the heatmap. The feature attack of [61] generates an image similar to the most activating training image which adds less new information. In contrast, our NPFV allows to identify which features the component has picked up.



Figure 9: Illustration of the information shown for labeling neural PCA components: The illustration shows the visualization of the first neural PCA component of the class "mountain bike". The first image on the left shows the NPFV, the prediction of the robust ResNet50, its probability for the class "mountain bike", and the corresponding value of  $\alpha_l^{(k)}$ . The other five images shown, along with the corresponding probabilities and  $\alpha_l^{(k)}$ , are the maximally activating training images of this component. The second row shows GradCAM heatmaps with respect to the component  $\alpha_l^{(k)}(x)$ . Below the visualization, the labeler can select one of the two possible labels (*spurious* and *not spurious*) and navigate through the next or last neural PCA component.



Figure 10: **Illustration of the information shown for labeling images to create our "Spurious ImageNet":** This screenshot illustrates a tool that we used to create labels for our dataset and an example of the image that is chosen to be in our dataset, in class "hummingbird", as it contains the spurious feature bird feeder of the class "hummingbird" but no hummingbird.

features and contain no class features) for each image as can be seen in the Fig. 10;

• for each component, 75 images that are guaranteed to contain the spurious feature but not the class object of class k were selected by two human labelers. Images

were only accepted into the dataset if both labelers assigned the label "out".

# C. SpuFix

#### C.1. SpuFix - Orthogonal projection onto a nonorthogonal basis

Let  $L = |\mathcal{S}_k|$ . The projection can be written as a least squares problem: Let  $b_1, \ldots, b_L$  be the matched directions and  $B \in \mathbb{R}^{\tilde{D} \times L}$  the matrix containing them as columns. Now, the projection onto the subspace spanned by the  $b_l, l \in \{1, \ldots, L\}$  is given by

$$\min_{P^{(k)}\in\mathbb{R}^L} \left\|\tilde{\psi}_k(x) - \bar{\tilde{\psi}}_k - BP^{(k)}\right\|_2^2 \tag{13}$$

with closed-form solution

$$P^{(k)}(x) = (B^T B)^{-1} B^T \left( \tilde{\psi}_k(x) - \bar{\tilde{\psi}}_k \right).$$
(14)

## C.2. SpuFix - Recovering the original method

When using our robust ResNet50, it holds  $\tilde{f} = f$ . Then the matched directions are

$$b_{l}^{*} = \frac{\sum_{s \in I_{k}} \left( \psi_{k}(x_{s}) - \bar{\psi}_{k} \right) \alpha_{l}^{(k)}(x_{s})}{\left\| \sum_{s \in I_{k}} \left( \psi_{k}(x_{s}) - \bar{\psi}_{k} \right) \alpha_{l}^{(k)}(x_{s}) \right\|_{2}}$$
(15)

$$= \frac{Cv_l \langle \mathbf{1}, v_l \rangle}{\left\| Cv_l \langle \mathbf{1}, v_l \rangle \right\|_2} \tag{16}$$

$$= \frac{\lambda_l v_l \langle \mathbf{1}, v_l \rangle}{\left\| \lambda_l v_l \langle \mathbf{1}, v_l \rangle \right\|_2} = v_l \tag{17}$$

where  $\lambda_l$  is the eigenvalue corresponding to the eigenvector  $v_l$  and we have used:

$$\sum_{s \in I_k} \left( \psi_k(x_s) - \bar{\psi}_k \right) \alpha_l^{(k)}(x_s)$$
  
= 
$$\sum_{s \in I_k} \left( \psi_k(x_s) - \bar{\psi}_k \right) \left\langle \psi_k(x_s) - \bar{\psi}_k, v_l \right\rangle \left\langle v_l, \mathbf{1} \right\rangle$$
  
= 
$$Cv_l \left\langle v_l, \mathbf{1} \right\rangle$$

Thus, as the  $v_l$  are an orthonormal basis it holds  $P_l^{(k)}(x) = \langle v_l, \tilde{\psi}_k(x) - \bar{\psi}_k \rangle$  and we get

$$\tilde{f}_k^{SpuFix}(x) \tag{18}$$

$$= \tilde{f}_k(x) - \sum_{l \in \mathcal{S}_k} \max\{\langle \mathbf{1}, v_l \rangle P_l^{(k)}(x), 0\}$$
(19)

$$=f_k(x) - \sum_{l \in S_k} \max\{\alpha_l^{(k)}, 0\} = f_k^{SpuFix}(x).$$
(20)

#### **D.** Quantitative Evaluation

In this section, we extend the quantitative results given in the main paper in Tab. 1 for a large number of ImageNet models. In Tab. 2 we show ILSVRC-2012 test accuracies and the mean spurious AUC (mAUC) for a wide selection of ImageNet models with different architectures and training configurations. Again, our spurious AUC is computed classwise using the predicted probability for that class as a score where we compare the images corresponding to this class of "Spurious ImageNet" (not showing the class object, but just the spurious feature) vs the ImageNet validation set images of that class. Finally, we take the mean of all classwise AUCs to get the final mAUC. All models except for our multiple-norm robust ResNet50 and the robust ResNet50 from [61] are taken from PyTorch Image Models [77].We further distinguish between models trained on ImageNet1k only (in1k), models pre-trained on ImageNet21k and then fine-tuned on ImageNet1k (in1kFT21k), ImageNet21k classifiers (in21k) and models trained using semi-supervised training techniques on large datasets containing 100M or more images [81, 84] and multimodal CLIP [49] models that are pre-trained on large datasets containing text and image pairings [55, 54]. Note that we do not report accuracies for ImageNet21k models as no test set for in21k is available. All models pre-trained on other datasets than ImageNet21k are Imagenet 1k models with a classification head containing 1000 classes after potential fine-tuning. In Fig. 11, we also plot mean spurious AUC against ImageNet-1k test accuracy and color code the models based on the dataset used during training.

Pre-training and fine-tuning: Overall, the trend seems to be that better models (in terms of accuracy) improve in mAUC and are less vulnerable to spurious features. It is also easily observable that pre-training on larger datasets such as ImageNet-21k can help to decrease vulnerability to spurious features, which can be seen best from the EfficientNetv2-M/L, ViT-B/L AugReg and ConvNeXt-L models for which we can evaluate the difference between in1k, in1kFT21k, and in21k training. The in1k ViT-B16AugReg achieves an mAUC of 0.850 whereas the same in21k model achieves an mAUC of 0.931 before and 0.917 after in1k fine-tuning. Similar trends are also visible for the EfficientNetv2-M and ConvNeXt-L models, where all ImageNet21k models (in1kFT21k and in21k) perform better than pure in1k models, however, parts of the improvement of the in21k models is lost during fine-tuning. While we do not have pure in1k models for them to compare to, other in21k pre-trained models such as the Big Transfer models, as well as the standard ViT's without AugReg, the BEiT and Swin architecture-based models show the same behavior and decrease spurious mAUC during fine-tuning. It thus remains an open question how one can use the benefits of pre-training on massive datasets with fine-grained

	Orig	inal	SpuFix			Original		SpuFix	
Name	Acc.	Spu.	Acc.	Spu.	Name	Acc.	Spu.	Acc.	Spu.
ImageNet1k			1B	1B Instagram[84]					
Rob. ResNet50	57.4%	0.630	56.8%	0.763	ResNeXt101 SSL [84]	83.3%	0.872	83.3%	0.875
Rob. ResNet50[61]	57.9%	0.651	57.2%	0.764	ResNeXt50 SSL [84]	82.2%	0.857	82.1%	0.862
ResNet50[29]	81.2%	0.851	81.2%	0.860	ResNet50 SSL [84]	81.2%	0.850	80.8%	0.865
ResNet101[29]	82.8%	0.748	82.8%	0.795	Ima	geNet21k	FT1k		
ResNeXt50 32x4d[82]	82.0%	0.783	82.0%	0.808	ResNetV2-152 BiT[36]	84.9%	0.895	84.9%	0.900
ResNeXt101 32x8d[82]	79.3%	0.797	79.2%	0.811	ResNetV2-50 BiT[36]	84.0%	0.887	84.0%	0.895
ResNeXt101 64x4d[82]	83.2%	0.779	83.2%	0.786	EfficientNetV2-M[70]	86.0%	0.892	86.0%	0.897
EfficientNet B5 RA[17]	83.8%	0.829	83.8%	0.833	EfficientNetV2-L[70]	86.8%	0.893	86.8%	0.898
EfficientNet B5 AP[83]	84.3%	0.828	84.2%	0.832	ConvNeXt-B[41]	86.3%	0.892	86.3%	0.895
EfficientNet B6 AA[69]	84.1%	0.830	84.1%	0.836	ConvNeXt-L[41]	87.0%	0.910	87.0%	0.913
EfficientNet B6 AP[83]	84.8%	0.831	84.8%	0.838	ConvNeXt-XL[41]	87.3%	0.908	87.3%	0.913
EfficientNet B7 RA[17]	84.9%	0.834	84.9%	0.839	ConvNeXtV2-B[79]	87.6%	0.907	87.6%	0.911
EfficientNet B7 AP[83]	85.1%	0.826	85.1%	0.831	ConvNeXtV2-L[79]	88.2%	0.905	88.2%	0.907
EfficientNetV2-M[70]	85.2%	0.846	85.2%	0.856	ConvNeXtV2-H[79]	88.7%	0.919	88.7%	0.923
EfficientNetV2-L[70]	85.7%	0.851	85.7%	0.860	DeiT3-S\16[72]	83.1%	0.845	83.1%	0.860
ConvNeXt-B[41]	84.4%	0.802	84.4%	0.816	DeiT3-L\16[72]	87.7%	0.895	87.7%	0.901
ConvNeXt-L[41]	84.8%	0.803	84.8%	0.819	Swin-B 224[40]	85.3%	0.877	85.3%	0.883
ConvNeXtV2-B[79]	85.5%	0.848	85.5%	0.856	Swin-L 384[40]	87.1%	0.898	87.1%	0.901
ConvNeXtV2-L[79]	86.1%	0.845	86.1%	0.858	SwinV2-L[39]	87.5%	0.889	87.5%	0.891
ConvNeXtV2-H[79]	86.6%	0.867	86.6%	0.879	ViT-B\16 224	81.8%	0.881	81.7%	0.889
DeiT3-S\16 224[72]	81.4%	0.851	81.4%	0.859	ViT-B\16 384[20]	84.2%	0.905	84.2%	0.912
DeiT3-L\16 384[72]	85.8%	0.863	85.8%	0.877	ViT-L\16 †	85.8%	0.914	85.8%	0.923
ViT-B\16 †	81.1%	0.850	81.1%	0.859	ViT-B\16 †	86.0%	0.917	85.9%	0.925
VOLO-D5 512[85]	87.1%	0.882	87.1%	0.907	BEiT-B\16 224[8]	85.2%	0.890	85.2%	0.897
VOLO-D5 224[85]	85.4%	0.863	85.3%	0.890	BEiT-L\16[8]	88.6%	0.921	88.6%	0.927
JFT-300M[28]			BEiTV2-L\16 224[46]	88.4%	0.921	88.4%	0.925		
EfficientNet B5 NS [81]	86.1%	0.924	86.1%	0.924	I	mageNet2	1k	I	1
EfficientNet B6 NS [81]	86.5%	0.875	86.5%	0.880	ResNetV2-152 BiT[36]	-	0.908	-	0.908
EfficientNet B7 NS [81]	86.8%	0.907	86.9%	0.912	ResNetV2-50 BiT[36]	-	0.910	-	0.910
EfficientNet L2 NS [81]	88.4%	0.914	88.3%	0.917	EfficientNetV2-M[70]	-	0.919	-	0.919
Y	FFC-100N	Λ			EfficientNetV2-L[70]	-	0.929	-	0.929
ResNeXt101 SSL [84]	81.8%	0.833	81.8%	0.841	ConvNeXt-B[41]	-	0.939	-	0.939
ResNeXt50 SSL [84]	80.3%	0.821	80.2%	0.831	ConvNeXt-L[41]	-	0.943	-	0.943
ResNet50 SSL [84]	79.2%	0.804	78.8%	0.828	ConvNeXt-XL[41]	-	0.945	-	0.945
LA	ION-2B[5	54]			Swin-B 224[40]	-	0.808	-	0.808
CNeXt-B CLIP[49] †	86.2%	0.859	86.2%	0.865	Swin-L 384[40]	-	0.820	-	0.820
CNeXt-L CLIP[49] † 224	87.3%	0.858	87.3%	0.865	ViT-L\16 †	-	0.931	-	0.931
CNeXt-L CLIP[49] † 384	87.8%	0.879	87.9%	0.884	ViT-B\8 †	-	0.931	-	0.931
ViT-L\14 CLIP[49] 336	88.2%	0.912	88.2%	0.914	BEiT-B\16 224[8]	-	0.935	-	0.935
LAI	DN-400M	[55]		·	BEiT-L\16 224[8]	-	0.940	-	0.940
EVA-G\14 CLIP 336[24]	89.5%	0.911	89.4%	0.915	BEiTV2-L\16 224[46]	-	0.951	-	0.951
	·								
EVA-G\14 CLIP 560[24]	89.8%	0.919	89.8%	0.925					

Table 2: Extended version of Tab. 1 from the main paper. We show ImageNet1k Accuracy (Acc.) and mean spurious AUC (Spu.) for the original model and the SpuFix version for a wide selection of state-of-the-art ImageNet classifiers, trained on: either ImageNet1k only (ImageNet1k), pre-trained on ImageNet21k and then fine-tuned on ImageNet1k (ImageNet21kFT1k), full ImageNet21k classifiers (ImageNet21k) or pre-training on a range of other datasets (JFT-300M, YFFC-100M, LAION-2B, LAION-400M, MIM, 1B Instagram). For models commonly used with different input resolutions, we state the used one at the end of the name. Models using AugReg[66] are marked with †. The ResNext50 and ResNext101 trained with SSL [84] have cardinality 32 and group width 4 and 16, respectively.



Figure 11: We plot Accuracy versus mean spurious AUC for a wide variety of SOTA ImageNet classifiers. Models that use the same architecture family use the same marker and we use color coding for the (pre-taining) datasets. For example, all models that are pre-trained on ImageNet21k and then fine-tuned are marked in yellow whereas standard ImageNet1k models are marked green. As can be observed, the addition of larger datasets like ImageNet21k, JFT-300M or LAION does decrease vulnerability to spurious features over ImageNet1k models with comparable accuracy. The arrows show the consistent improvement of mean spurious AUC after applying SpuFix while the change of accuracy is negligible for most of the models.

class structures to preserve or even improve mAUC during fine-tuning to smaller datasets such as ImageNet1k.

Different architectures: In terms of architecture, there is no easily observable trend. On pure in1k models, VOLO D5 achieves the best mAUC of 0.882, however, it is also the most accurate model. The best overall model in terms of mAUC is the BEiTV2-L in21k with an mAUC of 0.951, however, after fine-tuning, the mAUC decreases to 0.921 where it achieves similar values as some other models like the ViT-B Augreg (0.917), ConvNextV2-H (0.919) and BEiT-L (0.921) which are the best models with ImageNet-1k classification head in terms of mAUC. In summary, attention-based transformers do not seem to yield strong benefits over convolutional neural networks in terms of vulnerability to spurious features. From Table 2, we also see that semi-supervised training approaches like Noisy Student self-training[81] can help to improve mAUC over pure ImageNet-1k training. However, there the smallest EfficientNet B5 actually achieves better mAUC than all other models, even the EfficientNet L2 which achieves much better clean accuracy. Pre-training using CLIP on large image/text datasets can also yield models with mAUC above 0.9 and is comparable to in21k pre-training. For example, the ViT-L achieves an mAUC of 0.914 (with AugReg) after in21k pre-training and 0.912 after CLIP pre-training on LAION-2B (without AugReg).

SpuFix on the robust ResNet50: The robust ResNet50 shows a substantial improvement in mAUC from 0.630 to 0.763. Fig. 13 to 15 show the class-wise values. In particular, it raises the class-wise spurious AUC from 0.332 to 0.932 for bookshop (+60.0%), from 0.279 to 0.819 for flagpole (+54.0%) and from 0.246 to 0.778 for Band Aid (+53.3%). Overall, the mAUC increases for 95 of 100 classes and achieves an improvement of at least 0.1 for 49 of them. Both the SpuFix method and the image collection procedure for the Spurious ImageNet benchmark are based on the values  $\alpha_l^{(k)}$  for spurious NPCA components l. Thus, to further validate the benefit of SpuFix, we collected 10 images each for the classes hummingbird, gondola and flagpole containing only the spurious feature (bird feeder, building/canal, US flag) without any automated filtering, i.e. neither model predictions nor NPCA components were used. Fig. 12 shows the predictions as well as the class probability for the spuriously correlated class of the original model and the SpuFix version for these images. The harmful predictions, i.e. predictions of the spuriously correlated class, are reduced from 6 to 3 for hummingbird, 8 to 4 for gondola and 7 to 0 for flagpole. SpuFix also decreases the mean class probability over the 10 images: from 0.57 to 0.13 (hummingbird), 0.61 to 0.13 (gondola), and 0.70 to 0.05 (flagpole). The actual improvements for the individual images are even larger due to the fact that the original model already does not predict the corresponding class or shows a low class probability for some of them. This shows that SpuFix indeed mitigates the reliance on these spurious features independent of the image collection procedure.

SpuFix on other ImageNet classifiers: In addition to the values for the original models, Tab. 2 also contains the mAUC and accuracies for the SpuFix versions of all evaluated models. Furthermore, the improvements are depicted as arrows in Fig. 11. One can see that SpuFix consistently improves the mAUC consistently for all models that were trained or fine-tuned on ImageNet1k. Even the top performing models still benefit significantly, e.g. 0.919 to 0.925 (+0.6%) for the EVA-Giant\14 CLIP 560 (MIM) or 0.921 to 0.927 (+0.6%) for the BEiT-L\16 pre-trained on ImageNet21k. On the other hand, the effect of SpuFix on the validation accuracy is negligible. Only the different variants of the ResNet50 architecture show a decrease of more than 0.1%. However, these models also achieve the largest improvements in spurious mAUC, e.g. the ResNet50 SSL (1B Instagram) loses 0.4% accuracy but also has a significant gain of +1.5% in spurious mAUC. We want to stress again that SpuFix can be applied to any ImageNet classifier with minimal effort and the code for doing so is part of the github repo (no retraining, labels etc required).

Models with high mAUC still rely on harmful spurious features: While the general trend of accuracy versus mAUC validates the progress of recent vision models, it does not mean that spurious features are no longer a problem for those models, especially since the worst-performing classes are still severe modes of failure. To better understand this behavior on individual classes, we plot the class-wise AUC for all 100 classes and a selection of models in Fig. 13 to 15. While both robust ResNet50 models are overall worse than the much larger comparison models ViT-B AugReg, we highlight that our SpuFix method (see Section 7.3) does significantly improve the mean spurious AUC of both ResNets50 models without requiring retraining. The ViT-AugReg also benefits from SpuFix but due to the transfer to smaller extent - nevertheless one can observe improvements by more than 5% in AUC for the classes flagpole, pole, puck, bookshop, lighter. On average, it is again observable that ImageNet-21k pre-training does improve spurious AUC. However, in terms of the final ImageNet-1k classifier after fine-tuning, classes like bakery, flagpole, wing, or pole remain challenging and can have a class-wise AUC as low as 0.5. Thus the improvements seem to depend heavily on the structure of the dataset used for pretraining and whether or not this dataset contains the spurious feature as an individually labeled class that allows the model to distinguish the class object from the spurious feature. For example, ImageNet-21k contains both flagpole and flag as separate classes, thus in21k ViT-B model achieves much better spurious AUC for these class (Spurious ImageNet contains flag images without flagpoles) than



Figure 12: Validation of SpuFix independent of Spurious ImageNet: We collected 10 images each for the classes hummingbird/gondola/flagpole containing only the spurious feature (bird feeder/building/US flag) without filtering by model predictions or  $\alpha_l^{(k)}$ . Our robust ResNet50 classifies 6/8/7 as the corresponding class (mean class probability 0.57/0.61/0.70), the SpuFix version only 3/4/0 (mean class probability 0.13/0.13/0.05). Therefore, SpuFix reduces the reliance on these harmful spurious features independent of the image collection procedure.

the ViT-B with a ImageNet-1k classification head. Nevertheless, even the in21k models still show a low AUC for flag pole and thus have problems distinguishing between flags (spurious feature) and flag pole. It also has to be noticed that the seemingly high AUC values are sometimes misleading. First of all, we stress again that the images of Spurious ImageNet do not contain the class object and thus an AUC of one should be easily obtainable for a classifier. Second, even if the AUC is one, it only means that the predicted probability for the validation set images (containing the class object) is always higher than the predicted probability for images from Spurious ImageNet (not containing the class object). However, still, a large fraction of the Spurious ImageNet images can be classified as the corresponding class, e.g. the ConvNext-L-1kFT21k has a class-wise AUC of 0.93 for "quill", but still 71% of all Images from Spurious ImageNet are classified as "quill", see Fig. 16 where we show for each image from "Spurious ImageNet" the top-3 predictions with their predicted probabilities. Thus the class extension can still be significant even for such a strong model. There are also classes which are completely broken like puck with an AUC of 0.69, where 100% of all images in "Spurious ImageNet" are classified as puck. The reason is that the puck is simply too small in the image (or sometimes even not visible at all), whereas the ice hockey players and also part of the playing field boundary are the main objects in the image. Thus the classification is only based on spurious features and the object "puck" has never been learned at all.

#### E. Comparison to Neural Features of [61]

In this section, we quantitatively and qualitatively evaluate the diversity of the subpopulations detected by our top-5 NPCA components and the top-5 neurons of [61], respectively, on all 1000 classes. To enable a direct comparison, we consider the NPCA components computed on their robust ResNet-50 [61] and compare them to the top-5 neurons detected in [61] for the same model.

A larger variety of subpopulations increases annotation efficiency as duplicates of the same semantic feature do not add more information. Moreover, the probability of missing a (harmful) spurious feature is higher when several of the top components/neurons correspond to the same feature because it might drop out of the set selected for human supervision.

For the quantitative evaluation, we measure the perceptual similarity of the subpopulations based on the *matched distances* of the maximally activating training images:

Let  $I_i^{(k)}$  be the set of the maximally activating images of component *i* for class *k* and let *d* be a perceptual metric. We define the *matched distance*  $d_m\left(x, I_j^{(k)}\right)$  of an image  $x \in I_i^{(k)}$  to a component  $I_j^{(k)}, j \neq i$ , as the minimal (perceptual) distance between x and the five images in  $I_i^{(k)}$ :

$$d_m\left(x, I_j^{(k)}\right) := \min_{x' \in I_j^{(k)}} d(x, x').$$

For every class k, we consider the top-5 components/neurons, each represented by the 5 maximally activating training images of the corresponding class k. We compute the matched distances of every image to the other four components, resulting in a total of 100000 matched distances. Fig. 17 shows histograms for the perceptual metrics LPIPS [87], 12-distance of the CLIP embeddings [23], and the SSCD distance [47]. In all three of them, the distribution corresponding to the NPCA components is shifted to the right compared to the one of the neural features which supports the hypothesis that NPCA detects more diverse subpopulations. However, as the purpose of the available perceptual metrics is to measure perturbations of the same image [87] or to detect (close) copies [47], there is no guarantee that these distances are still meaningful for larger values. Nevertheless, the maximally activating training images of the top-5 neurons of [61] also have a larger amount of distances equal to zero which corresponds to identical images. The histogram in Fig. 18 shows how many of these identical pairs occur per class. For the NPCA, identical maximally training images occur less often and for most of the cases there is only one per class. The method of [61] produces several identical images per class much more frequently which confirms that single neurons do not necessarily capture different concepts. Due to the orthogonality constraint of PCA, the NPCA components explore different directions in feature space and detect subpopulations with less overlap.

To visualize these results, Fig. 19 and Fig. 20 show the three worst classes with respect to identical maximally activating training images for [61] and NPCA, respectively. In both figures, we show the 5 maximally training images together with their GradCam images per top-5 neuron of [61] on the left resp. top-5 NPCA component on the right.

Worst three classes – [61] – (28/24/24 identical pairs): Considering the classes "badger" and "king snake", the five neurons capture almost equivalent semantic features which are all labeled as "core". For "badger", three neurons have exactly the same 5 maximally activating images. A similar pattern holds for the class "groom". Here, four of the subpopulations found by [61] are very similar. Note that three of those were labeled as "spurious feature" and one as "core feature". This is a consequence of the use of a majority vote of the human labelers as a selection criterium. Our stricter criterium (unanimous vote) prevents such inconsistencies. While, in all three cases, one of the NPCA components ("badger": first comp., "king snake": first comp., "groom": third comp.) resembles the features detected by the neurons, the remaining components capture a much more di-



Figure 13: Extended version of Fig. 6 from the main paper for the first 35 classes in our dataset. We plot class-wise spurious AUC for our robust ResNet50, the robust ResNet50 from [61], and a ViT-B, both trained on ImageNet1k with and without pre-training on ImageNet21k as well as pure ImageNet21k training. Additionally, we show the corresponding SpuFix versions of the five models.



Figure 14: Continued from Fig. 13 for classes 36-70.



Figure 15: Continued from Fig. 13 for classes 71-100.

quill: 0.97 book jacket: 0.01 fountain pen: 0.00 Martin and the second and the second and the Martin and Second Martin a	quill: 0.92 envelope: 0.04 fountain pen: 0.01 n someone treats you at ption, help them narrow ea by removing yourself requation. It's that simp Longitudine. Nate	quill: 0.91 book jacket: 0.02 fountain pen: 0.01	quill: 0.89 fountain pen: 0.01 packet: 0.01 Great 2017 The second	quill: 0.88 fountain pen: 0.08 envelope: 0.01	quill: 0.85 envelope: 0.04 fountain pen: 0.03 for the same pair of the same for the same pair of the same for the same pair of the same for the same pair of the same pair of the same pair of the same pair of the	quill: 0.83 fountain pen: 0.05 book jacket: 0.01 Curr Acta Stopp H. Chang and State Stopp H. Chang and State Stopp H. Chang and State Stopp Hange Ka- bard State Stopp Hange Ka- shall at the State St	quill: 0.81 envelope: 0.08 fountain pen: 0.01 just pen in spirit fast dang a world gue arisk jusu hat de re of? Genetationedeus int more time [Jamily and made more a	quill: 0.79 book jacket: 0.07 fountain pen: 0.03 blog significant and and an blog significant an blog significant an blog significant an blog	quill: 0.79 envelope: 0.07 fountain per: 0.03 Martin and the second second second second seco
quill: 0.78 book jacket: 0.07 fountain pen: 0.03 dr. FER ANSACETY Mark and ANSACETY and ANSACETY ANSACETY and ANSACETY A	quill: 0.78 book jacket: 0.04 envelope: 0.03 ptus: futy 20 fut imum in lucem editus upletifitmo indice d	quill: 0.78 envelope: 0.05 book jacket: 0.03 menoisti and the file of the file book of the file book of the file book of the file book of the file of	quill: 0.77 book jacket: 0.08 envelope: 0.05 "The design of the second "The design of the second "The design of the second "The design of the second second of the second of the second of the second of the second is status of the second of the second of the second is status of the second of the second of the second is status of the second of the second of the second is status of the second of the second of the second of the second is status of the second of the sec	quill: 0.72 envelope: 0.09 menu: 0.02 we de Otta, monomos Reg u d'för mortal parches s moffe da pune e non ha stin weato non ha la guerra pe non puis supre de che in un ha filser i un ui ha rota non	quill: 0.71 book jacket: 0.04 prayer rug: 0.02	quill: 0.69 fountain pen: 0.06 book jacket: 0.02	quill: 0.67 book jacket: 0.09 envelope: 0.02	guill: 0.65 fountain pen: 0.10 book jacket: 0.06	quill: 0.65 book jacket: 0.13 fountain pen: 0.04
quill: 0.64 envelope: 0.08 book jacket: 0.04	quill: 0.64 book jacket: 0.06 fountain pen: 0.03	quill: 0.61 envelope: 0.04 fountain pen: 0.03 branchave if herman enveloper: Jermen table for may Seeme table for may Seeme table for may Herme in hoge: Stoppin Harking	quill: 0.61 book jacket: 0.08 fountain pen: 0.08 We have been the second	guill: 0.59 fountain pen: 0.23 book jacket: 0.02	quill: 0.56 fountain pen: 0.09 binder: 0.07	quill: 0.55 envelope: 0.21 book jacket: 0.04 Star defe (free fail of the second of the	quill: 0.54 fountain pen: 0.15 binder: 0.02	quill: 0.52 fountain pen: 0.25 binder: 0.04	quill: 0.51 fountain per: 0.17 book jacket: 0.08
quill: 0.50 book jacket: 0.15 fountain per: 0.12	quill: 0.48 fountain pen: 0.31 binder: 0.03	quill: 0.48 book jacket: 0.14 fountain pen: 0.10	quill: 0.47 envelope: 0.27 book jacket: 0.08	<section-header></section-header>	quill: 0.44 book jacket: 0.37 contain per: 0.01 meter of the second seco	quill: 0.44 tray: 0.34 book jacket: 0.03		quill: 0.44 envelope: 0.12 fountain pen: 0.10	quill: 0.43 book jacket: 0.26 fountain pen: 0.04
quill: 0.42 envelope: 0.09 book jacket: 0.06 Mit friedware Wolf an en- dicate from and the first careform and and the first careform and and the first careform and and the first framework of the first of the first of the first of the first	quill: 0.42 book jacket: 0.32 fountain pen: 0.04 contain pen: 0.04	quill: 0.42 fountain pen: 0.09 candle: 0.08 Collema for star to Italine to Stop allo do to for all Otlaw you to the star of Ulaw you the star of Ulaw you	quill: 0.41 fountain pen: 0.07 book jacket: 0.04	quill: 0.39 envelope: 0.16 slide rule: 0.07	quill: 0.39 book jacket: 0.36 fountain pen: 0.02		quill: 0.37 fountain pen: 0.21 binder: 0.04	quill: 0.37 fountain pen: 0.11 book jacket: 0.09	guill: 0.36 book jacket: 0.20 fountain pen: 0.12
quill: 0.34 fountain pen: 0.20 book jacket: 0.10	quill: 0.33 fountain pen: 0.31 binder: 0.05	quill: 0.31 book jacket: 0.24 fountain pen: 0.12	quill: 0.31 fountain pen: 0.16 book jacket: 0.15 sevent sevent se	quill: 0.30 binder: 0.13 fountain pen: 0.11 Martin de la constantina de la constanti	quill: 0.30 book jacket: 0.29 fountain pen: 0.05		quill: 0.24 fountain pen: 0.17 book jacket: 0.14	quill: 0.24 envelope: 0.13 book jacket: 0.11 k ndt in and the second second in a second second second second in a second second second in a second second second second second second in a second second second second second second second in a second second second second second second second second is second second is second s	guill: 0.23 fountain per: 0.19 book jacket: 0.12
quill: 0.21 book jacket: 0.18 fountain pen: 0.11	quill: 0.15 paper towel: 0.09 menu: 0.05 Karylat tala-dar waga, malan to nerapatan walan menganar di wana bashagi dapit dilitat nela hala. Maladadan	book jacket: 0.85 quill: 0.03 menu: 0.01 Menu: 0.01 Men	envelope: 0.68 quill: 0.16 web site: 0.01 DPZ Iver. monif.'s Restaurss news for Dis Spannis Dis IKA FOOD A SPECIALT west Key	book jacket: 0.66 quill: 0.16 binder: 0.01 with the second second second second with the second seco	book jacket: 0.59 quill: 0.28 menu: 0.01	book jacket: 0.58 quill: 0.27 fountain pen: 0.02	book jacket: 0.50 quill: 0.09 fountain pen: 0.08	walking stick: 0.49 quill: 0.23 fountain pen: 0.07	fountain pen: 0.42 quill: 0.33 book jacket: 0.07
book jacket: 0.37 quill: 0.27 whiskey jug: 0.05 Vege Corey Fonfes, Prefides Conforde in Julia Conforde	book jacket: 0.36 quill: 0.20 fountain pen: 0.09 ***********************************	book jacket: 0.34 whiskey jug: 0.19 quill: 0.06 PAR.13115, Benediction Presoft, in C Bunalo, wa Frementila, fab Stella antes 1551- M 2 Chr. Agen	book jacket: 0.33 quill: 0.23 fountain pen: 0.08	book jacket: 0.30 envelope: 0.13 quill: 0.11 matorum du ven statute ven pritta D MOENVM. batobanes spin. D. xerr M. M. Munffr					

Figure 16: We show all images of class "quill" (spurious feature: handwritten text) in Spurious ImageNet together with the top-3 predicted probablities of the ConvNext-L-1kFT21k. Despite a class-wise AUC of 0.93 which seemingly suggests that the spurious feature is not playing a big role anymore, we observe that 76% of the images are classified as "quill" despite no "quill" being present. Thus the spurious class extension is still present but the classifier produces slightly less confident predictions on these images.



Figure 17: **Histograms of matched distances:** We consider the 5 maximally activating training images for each of the top-5 NPCA components resp. top-5 neurons of [61]. For each of these images (and each of the components/neurons) we find the best matching maximally activating training image of a different component/neuron. We call the distances of the corresponding images "matched distance" and plot these for three different metrics: the neural perceptual metric [87], the 12-distance of clip embeddings [23], the SSCD distance used for image copy detection [47].



Figure 18: Histogram of identical maximal activating training images for the top-5 NPCA components resp. the top-5 neurons of [61] for the robust model used by [61] We observe that for NPCA only a few components have identical maximally activating training images and if it happens in the majority of cases only a single maximal activating training image of two components is identical. In contrast, [61] has a long tail, meaning that several maximal activativating training images of top-5 neurons are identical. This confirms that maximally activated neurons do not necessarily capture different semantic concepts. This is different for NPCA as the orthogonality constraint of PCA enforces to explore different directions/regions in feature space.

verse set of features. In the case of "king snake", the fourth NPCA components detects, with respect to the criteria of [61], a spurious feature (hands). For "groom", we even have three NPCA components corresponding to spurious features (bride, ceiling/lights, trees/bushes). This illustrates how a lack of diversity in the subpopulations of [61] for

some classes can hinder the detection of spurious features.

**Worst three classes – NPCA – (10/8/5 identical pairs):** First, we note that regarding the number of identical pairs for the "worst cases" of NPCA, there exist many classes for the top-5 neurons which have similar number of identical pairs.

The class "lumbermill" is the worst class for NPCA. One can see two pairs of duplicate subpopulations (trunks and trunks/planks) that overlap to large extent with each other (which is an absolute outlier for NPCA). However, interestingly this subpopulation which is clearly spurious for "lumbermill" as there are no particular features for "lumbermill" is not present in the top-5 neurons of [61]. The second worst class for NPCA is "barbershop" with two largely overlapping components showing store/house fronts. However, there are also three neurons that capture this semantic feature. In fact the number of identical pairs, NPCA 8, [61] 6, is not so different. In the case of "English foxhound", two NPCA components correspond to a white fence which is a spurious feature. While the neurons' subpopulations are unique for this class, they do not detect the spurious fence as GradCam for the neuron mainly activates on the dog.

Overall, these examples demonstrate that the problem of identical maximally activating images is a lot less severe for the NPCA components.

#### F. Extended Qualitative Evaluation

Here, we extend our qualitative evaluation of the found harmful spurious components from Figure 4. Concretely, for each such pair (class k, component l) we show in Fig. 21 and 22: i) random training images from class k, ii) NPFV of the component l together with the most activating images of  $\alpha_l^{(k)}$ , and iii) examples of images that display only the spurious feature but no class features and are **incorrectly** classified by four ImageNet classifiers as class k.



Figure 19: Classes, where [61] has the most identical pairs among the top-5 maximally activating images across different neurons (NPCA and neurons computed on the same model, Rob. ResNet50[61]). For each method we provide the number of identical pairs of images across different neurons for [61] resp. different components for NPCA as described in App. E. Each row shows the 5 maximally activating images together with the corresponding GradCAM heatmaps for the top-5 neurons of [61] (left) and the top-5 NPCA components (right), respectively. As in Fig. 5, our NPCA components are capturing different subpopulations in the training data for these classes, while the different neurons of [61] are finding many identical pairs, see App. E for more details.

**Note:** for these components, where [61] fails to find different subpopulations, our NPCA components find more diverse and even several spurious features: "hands" for the class "king snake" and "bride", "ceiling/lamps", and "trees/bushes" for the class "groom". The top-5 neurons of [61] for class "groom" identify three of the first four neurons as spurious and one as core even though the images are semantically the same and GradCAM activations are also similar. Semantically similar neurons are labeled differently due to their majority vote (for three of the neurons we have a 3:2 decision among the human labelers, for one a 4:1 decision), whereas we require that the two human labelers need to agree.

#### Lumbermill

Top-5 max. act. images of [61] ( <b>0 identical pairs</b> )	Top-5 max. act. images of NPCA (ours) (10 identical pairs)
Di in the second se	

Barbershop

Top-5 max. act. images of NPCA (ours) (8 identical pairs)

Top-5 max. act. images of NPCA (ours) (5 identical pairs)

DARDERS CON ADDRESS ACCOUNTS AND ADDRESS ACCOUNTS ADDRESS ADDRESS ACCOUNTS ADDRESS

**English foxhound** 

Top-5 max. act. images of [61] (1 identical pair)

Top-5 max. act. images of [61] (6 identical pairs)



Figure 20: Classes where NPCA has the most identical pairs among the top-5 maximally activating images across different NPCA components (NPCA and neurons computed on the same model, Rob. ResNet50[61]). For each method we provide the number of identical pairs of images across different neurons for [61] resp. different components for NPCA as described in App. E. Each row shows the 5 maximally activating images together with the corresponding GradCAM heatmaps for the top-5 neurons of [61] (left) and the top-5 NPCA components (right), respectively. While some of our NPCA components have identical pairs, the highest number of them (10) is almost three times smaller than the largest number of identical pairs of [61] (28). These examples show that even the worst classes for the NPCA components only contain a few overlapping subpopulations. This aligns with the observations in Fig. 18. Therefore, the problem of a lack of diversity in the detected features is much less severe for the NPCA components than for the neurons of [61].

# G. Random samples from our "Spurious ImageNet" dataset

To visualize our "Spurious ImageNet" dataset, for each of the 100 classes in our dataset, we show 4 randomly drawn images (out of the 75 in total) in Fig. 23 and 24. We also provide a label for the spurious feature shown in brackets. We again highlight that none of the images contains the actual class object.

# H. Generating the spurious feature to change predictions

In this section we show how one can adapt the recent method "Diffusion Visual Counterfactual Explanations" [5] to generate the spurious feature on a given image without changing the overall structure of the image. We first introduce the necessary notation. We denote by  $n(x) = \frac{x}{\|x\|_2}$  for  $x \neq 0$ , the normalization of a vector by its  $l_2$  norm and the confidence of the robust ResNet50 classifier in a target class k as

$$p_{\operatorname{robust},\psi}:[0,1]^d \to (0,1), \quad x \mapsto \frac{e^{f_{\operatorname{robust},\psi,k}(x)}}{\sum_{i=1}^K e^{f_{\operatorname{robust},\psi,i}(x)}}$$

Here,  $f_{\text{robust},\psi} : [0,1]^d \to \mathbb{R}^K$  are the logits of the robust classifier, and  $f_{\text{robust},\psi,k}(x)$  denotes the logit of class k.

To automatically add spurious features to any given image, we adapt a recently proposed method Diffusion Visual Counterfactual Explanations (DVCEs) [5], where at a step t the shifted mean  $\mu_t$  is of the form

$$g_{\text{update}} = C_c g_c - C_d g_d + C_a g_a,$$
  

$$\mu_t = \mu_\theta(x_t, t)$$
  

$$+ \Sigma_\theta(x_t, t) \|\mu_\theta(x_t, t)\|_2 g_{\text{update}},$$
  

$$p(x_{t-1}|x_t, \hat{x}, k) = \mathcal{N}(\mu_t, \Sigma_\theta(x_t, t)),$$

where  $g_c := n(\nabla_{x_t} \log p_{\text{robust},\psi}(k|f_{dn}(x_t,t)))$  is the normalized gradient of the adversarially robust classifier,  $g_d := n(\nabla_{x_t} d(\hat{x}, f_{\mathrm{dn}}(x_t, t)))$  - normalized gradient of the distance term. We add as additional guidance  $g_a :=$  $n(
abla_{x_t} lpha_j^{(k)}(f_{\mathrm{dn}}(x_t,t)))$  - the normalized gradient of the contribution  $\alpha_j^{(k)}$  of the *j*-th neural PCA component to the logit of class *k*. As the derivative of the diffusion models, relies on noisy updates, and the classifier has not been trained on such inputs, [5] propose to use the denoised sample  $\hat{x}_0 = f_{dn}(x,t)$  of the noisy input  $x_t$  as an input to the classifier. Intuitively, at every step t of the generative denoising process, the method of [5] follows i) the direction  $g_a$  that increases the contribution of neural PCA component j (corresponding to a desired spurious feature) of class k to the logit  $f_k(x)$  of this class, ii) the direction  $g_c$  that increases the confidence of the classifier in the class k, and iii) the direction  $g_d$  that decreases the distance to the original image  $\hat{x}.$ 

In our experiments, we set  $d(x, y) := ||x - y||_1$  following [5] and coefficients as follows:  $C_c = 0.1, C_d = 0.35, C_a = 0.05$ . With these parameters, we generate the desired DVCEs in Fig. 27. There, using minimal realistic perturbations to the original image we can change the prediction of the classifier in the target class k with high confidence. Moreover, these perturbations introduce only harmful spurious features to the image and not class-specific features e.g. for freight car the DVCE generates graffiti but no features of a freight car.

This happens, because, as has been shown qualitatively in Fig. 4 and quantitatively in Fig. 6, this classifier has learned to associate class "fireboat" with the spurious feature "water jet", "freight car" - with "graffiti", "flagpole" with a flag without the pole and mostly with "US flag", and "hard disc" - with "label", and therefore introducing only these *harmful* spurious features is enough to increase the confidence in the target class k significantly.



Figure 21: **Spurious features (ImageNet):** found by human labeling of our neural PCA components. For each class we show 5 random train. images (top left), the neural PCA Feature Visual. (NPFV) and 4 most activating train. images for the spurious feature component (bottom left). Right: four ImageNet models classify images **showing only the spurious feature but no class object** as this class.



Figure 22: **Spurious features (ImageNet):** found by human labeling of our neural PCA components. For each class we show 5 random train. images (top left), the neural PCA Feature Visual. (NPFV) and 4 most activating train. images for the spurious feature component (bottom left). Right: four ImageNet models classify images **showing only the spurious feature but no class object** as this class.



Figure 23: Random selection of 4 images for classes 1-25 of our "Spurious ImageNet" dataset with class label (spurious feature, NPCA component). Note that the labels of spurious features are coarse and thus overlap e.g. several are leaves/branches/fowers. We are not able to identify if these are special trees or flowers which might be more specific.



Figure 24: Random selection of 4 images for classes 26-50 of our "Spurious ImageNet" dataset with class label (spurious feature, NPCA component).



Figure 25: Random selection of four images for classes 51-75 of our "Spurious ImageNet" dataset with class label (spurious feature, NPCA component).



Figure 26: Random selection of 4 images for classes 76-100 of our "Spurious ImageNet" dataset with class label (spurious feature, NPCA component).



Figure 27: Adding spurious features automatically with an adaptation of DVCEs [5] changes the prediction of the classifier robust ResNet50. This happens, because, as has been shown qualitatively in Fig. 4 and quantitatively in Fig. 6, this classifier has learned to associate class "fireboat" with the spurious feature "water jet", "freight car" - with "graffiti", "flagpole" with a flag without the pole and mostly with "US flag", and "hard disc" - with "label". This again confirms that they are *harmful* spurious features.