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 $\alpha_l^{(k)}$ 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 $\alpha_l^{(k)}$ of the respective neural PCA component l;

- we have used the open-source tool[2] 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

[2]https://github.com/robertbrada/PyQt-image-annotation-tool

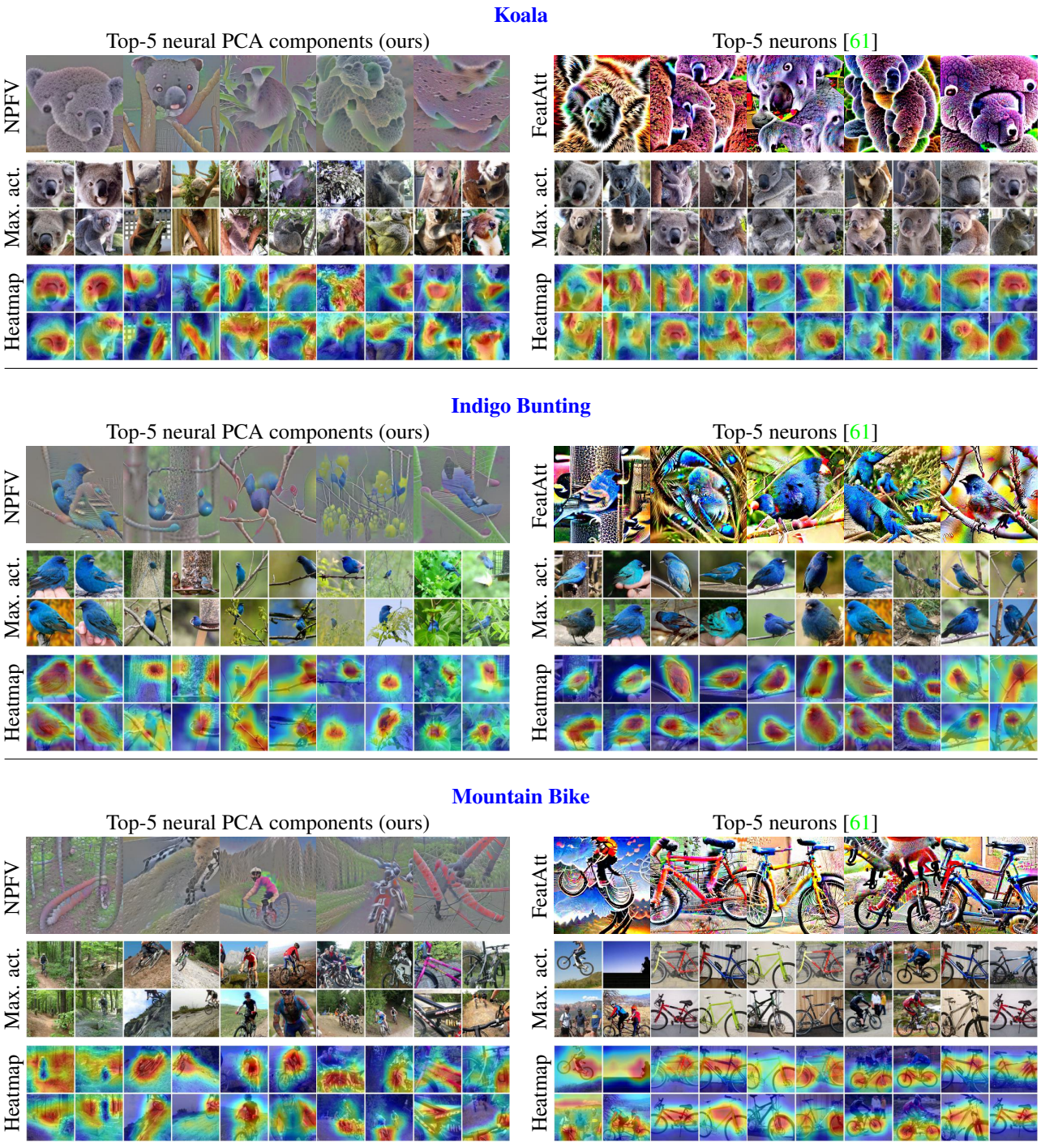

Figure 8: **Neural PCA feature components vs neural features of [61] for different classes (computed on our own multiple-norm robust model vs. their l_2-robust model):** First row: NPCA feature visualization (NPFV) of our top-5 NPCA components (left), and the feature attacks of the top-5 neurons of [61] (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.

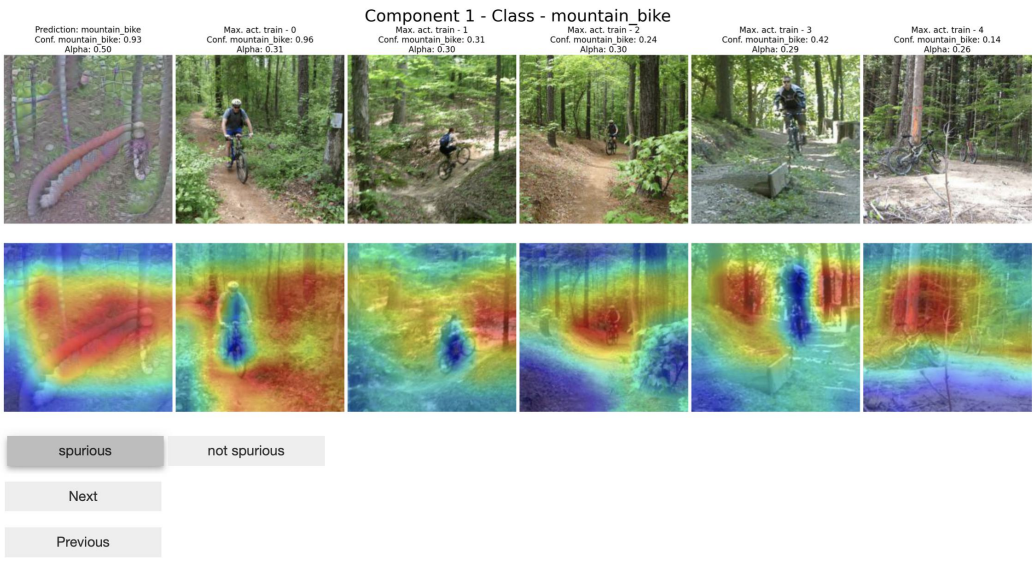

Figure 9: **Illustration of the information shown for labeling neural PCA components:** The illustration shows the visualization of the first neural PCA component of the class "mountain bike". The first image on the left shows the NPFV, the prediction of the robust ResNet50, its probability for the class "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.

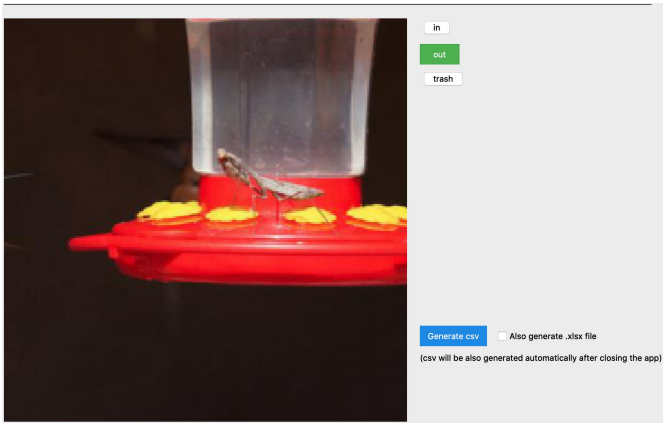

Figure 10: **Illustration of the information shown for labeling images to create our "Spurious ImageNet":** This screenshot illustrates a tool that we used to create labels for our dataset and an example of the image that is chosen to be in our dataset, in class "hummingbird", as it contains the spurious 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 non-orthogonal basis

Let $L = |\mathcal{S}_k|$. The projection can be written as a least squares problem: Let $b_1, \dots, b_L$ 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 $b_l, l \in \{1, \dots, L\}$ 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

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

C.2. SpuFix - Recovering the original method

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

$$b_l^* = \frac{\sum_{s \in I_k} \left(\psi_k(x_s) - \bar{\psi}_k \right) \alpha_l^{(k)}(x_s)}{\left\| \sum_{s \in I_k} \left(\psi_k(x_s) - \bar{\psi}_k \right) \alpha_l^{(k)}(x_s) \right\|_2} \tag{15}$$

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

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

where λ_l is the eigenvalue corresponding to the eigenvector v_l and we have used:

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

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

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

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

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

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-B\16 AugReg 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

Name	Original		SpuFix	
	Acc.	Spu.	Acc.	Spu.
ImageNet1k				
Rob. ResNet50	57.4%	0.630	56.8%	**0.763**
Rob. ResNet50[61]	57.9%	0.651	57.2%	**0.764**
ResNet50[29]	81.2%	0.851	81.2%	**0.860**
ResNet101[29]	82.8%	0.748	82.8%	**0.795**
ResNeXt50 32x4d[82]	82.0%	0.783	82.0%	**0.808**
ResNeXt101 32x8d[82]	79.3%	0.797	79.2%	**0.811**
ResNeXt101 64x4d[82]	83.2%	0.779	83.2%	**0.786**
EfficientNet B5 RA[17]	83.8%	0.829	83.8%	**0.833**
EfficientNet B5 AP[83]	84.3%	0.828	84.2%	**0.832**
EfficientNet B6 AA[69]	84.1%	0.830	84.1%	**0.836**
EfficientNet B6 AP[83]	84.8%	0.831	84.8%	**0.838**
EfficientNet B7 RA[17]	84.9%	0.834	84.9%	**0.839**
EfficientNet B7 AP[83]	85.1%	0.826	85.1%	**0.831**
EfficientNetV2-M[70]	85.2%	0.846	85.2%	**0.856**
EfficientNetV2-L[70]	85.7%	0.851	85.7%	**0.860**
ConvNeXt-B[41]	84.4%	0.802	84.4%	**0.816**
ConvNeXt-L[41]	84.8%	0.803	84.8%	**0.819**
ConvNeXtV2-B[79]	85.5%	0.848	85.5%	**0.856**
ConvNeXtV2-L[79]	86.1%	0.845	86.1%	**0.858**
ConvNeXtV2-H[79]	86.6%	0.867	86.6%	**0.879**
DeiT3-S\16 224[72]	81.4%	0.851	81.4%	**0.859**
DeiT3-L\16 384[72]	85.8%	0.863	85.8%	**0.877**
ViT-B\16 †	81.1%	0.850	81.1%	**0.859**
VOLO-D5 512[85]	87.1%	0.882	87.1%	**0.907**
VOLO-D5 224[85]	85.4%	0.863	85.3%	**0.890**
JFT-300M[28]				
EfficientNet B5 NS [81]	86.1%	0.924	86.1%	**0.924**
EfficientNet B6 NS [81]	86.5%	0.875	86.5%	**0.880**
EfficientNet B7 NS [81]	86.8%	0.907	86.9%	**0.912**
EfficientNet L2 NS [81]	88.4%	0.914	88.3%	**0.917**
YFFC-100M				
ResNeXt101 SSL [84]	81.8%	0.833	81.8%	**0.841**
ResNeXt50 SSL [84]	80.3%	0.821	80.2%	**0.831**
ResNet50 SSL [84]	79.2%	0.804	78.8%	**0.828**
LAION-2B[54]				
CNeXt-B CLIP[49] †	86.2%	0.859	86.2%	**0.865**
CNeXt-L CLIP[49] † 224	87.3%	0.858	87.3%	**0.865**
CNeXt-L CLIP[49] † 384	87.8%	0.879	87.9%	**0.884**
ViT-L\14 CLIP[49] 336	88.2%	0.912	88.2%	**0.914**
LAION-400M[55]				
EVA-G\14 CLIP 336[24]	89.5%	0.911	89.4%	**0.915**
MIM[24]				
EVA-G\14 CLIP 560[24]	89.8%	0.919	89.8%	**0.925**

Name	Original		SpuFix	
	Acc.	Spu.	Acc.	Spu.
1B Instagram[84]				
ResNeXt101 SSL [84]	83.3%	0.872	83.3%	**0.875**
ResNeXt50 SSL [84]	82.2%	0.857	82.1%	**0.862**
ResNet50 SSL [84]	81.2%	0.850	80.8%	**0.865**
ImageNet21kFT1k				
ResNetV2-152 BiT[36]	84.9%	0.895	84.9%	**0.900**
ResNetV2-50 BiT[36]	84.0%	0.887	84.0%	**0.895**
EfficientNetV2-M[70]	86.0%	0.892	86.0%	**0.897**
EfficientNetV2-L[70]	86.8%	0.893	86.8%	**0.898**
ConvNeXt-B[41]	86.3%	0.892	86.3%	**0.895**
ConvNeXt-L[41]	87.0%	0.910	87.0%	**0.913**
ConvNeXt-XL[41]	87.3%	0.908	87.3%	**0.913**
ConvNeXtV2-B[79]	87.6%	0.907	87.6%	**0.911**
ConvNeXtV2-L[79]	88.2%	0.905	88.2%	**0.907**
ConvNeXtV2-H[79]	88.7%	0.919	88.7%	**0.923**
DeiT3-S\16[72]	83.1%	0.845	83.1%	**0.860**
DeiT3-L\16[72]	87.7%	0.895	87.7%	**0.901**
Swin-B 224[40]	85.3%	0.877	85.3%	**0.883**
Swin-L 384[40]	87.1%	0.898	87.1%	**0.901**
SwinV2-L[39]	87.5%	0.889	87.5%	**0.891**
ViT-B\16 224	81.8%	0.881	81.7%	**0.889**
ViT-B\16 384[20]	84.2%	0.905	84.2%	**0.912**
ViT-L\16 †	85.8%	0.914	85.8%	**0.923**
ViT-B\16 †	86.0%	0.917	85.9%	**0.925**
BEiT-B\16 224[8]	85.2%	0.890	85.2%	**0.897**
BEiT-L\16[8]	88.6%	0.921	88.6%	**0.927**
BEiTV2-L\16 224[46]	88.4%	0.921	88.4%	**0.925**
ImageNet21k				
ResNetV2-152 BiT[36]	-	0.908	-	**0.908**
ResNetV2-50 BiT[36]	-	**0.910**	-	0.910
EfficientNetV2-M[70]	-	**0.919**	-	0.919
EfficientNetV2-L[70]	-	**0.929**	-	0.929
ConvNeXt-B[41]	-	0.939	-	**0.939**
ConvNeXt-L[41]	-	**0.943**	-	0.943
ConvNeXt-XL[41]	-	0.945	-	**0.945**
Swin-B 224[40]	-	0.808	-	**0.808**
Swin-L 384[40]	-	**0.820**	-	0.820
ViT-L\16 †	-	**0.931**	-	0.931
ViT-B\8 †	-	**0.931**	-	0.931
BEiT-B\16 224[8]	-	**0.935**	-	0.935
BEiT-L\16 224[8]	-	**0.940**	-	0.940
BEiTV2-L\16 224[46]	-	0.951	-	**0.951**

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.

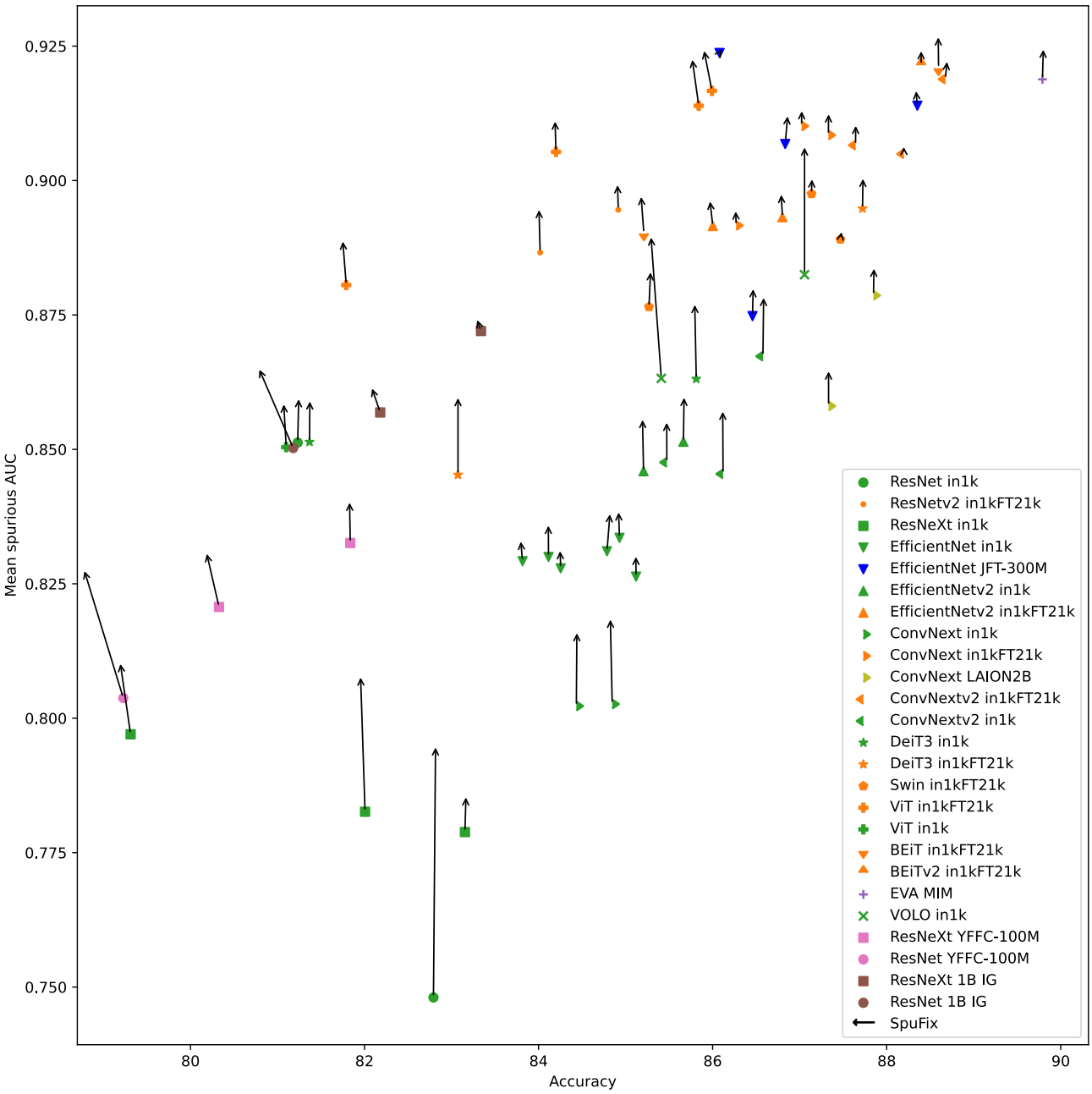

Figure 11: We plot Accuracy versus mean spurious AUC for a wide variety of SOTA ImageNet classifiers. Models that use the same architecture family use the same marker and we use color coding for the (pre-taining) datasets. For example, all models that are pre-trained on ImageNet21k and then fine-tuned 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 $I_j^{(k)}$:

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

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], l2-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-

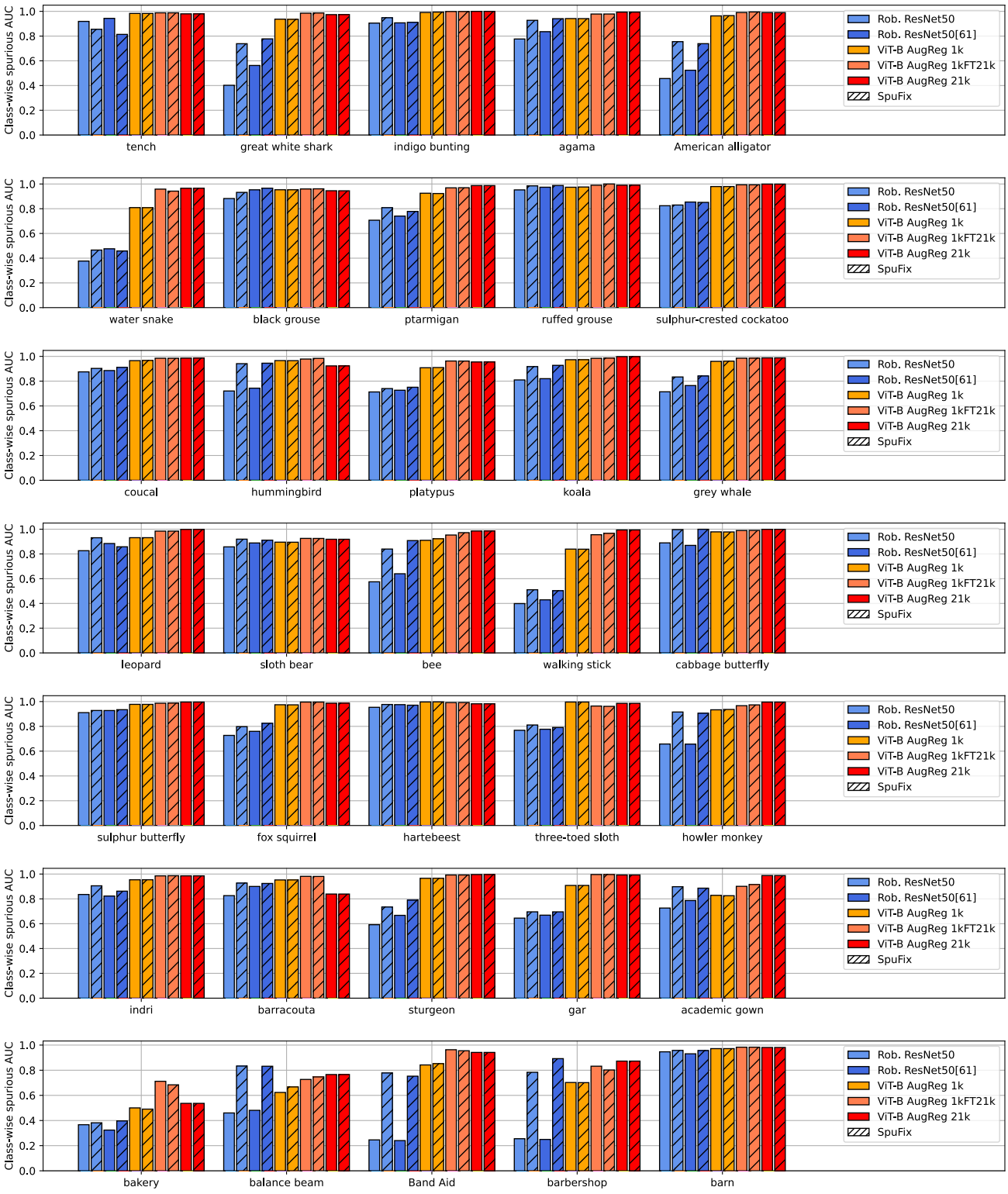

Figure 13: Extended version of Fig. 6 from the main paper for the first 35 classes in our dataset. We plot class-wise spurious AUC for our robust ResNet50, the robust ResNet50 from [61], and a ViT-B, both trained on ImageNet1k with and without pre-training on ImageNet21k as well as pure ImageNet21k training.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

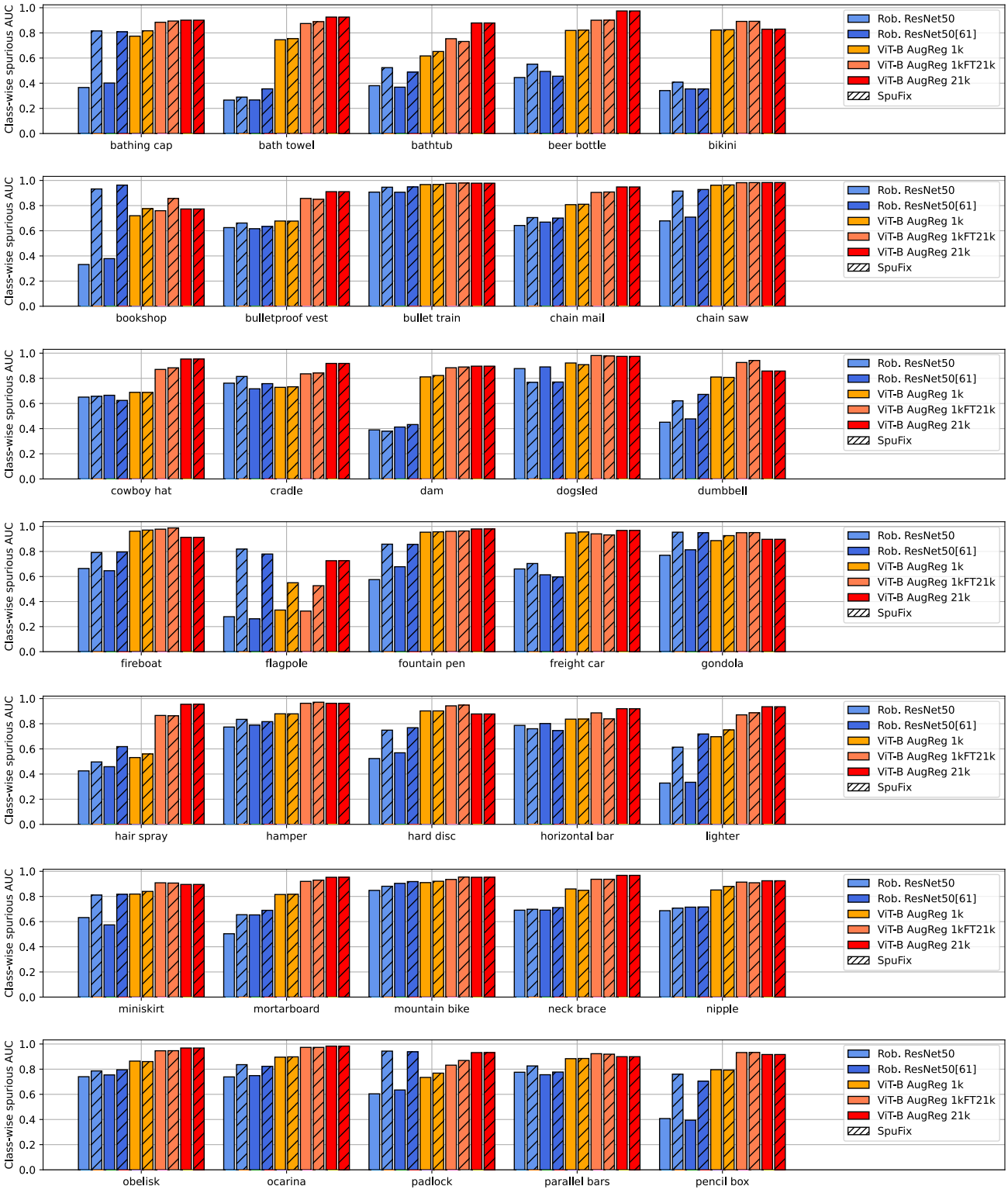

Figure 14: Continued from Fig. 13 for classes 36-70.

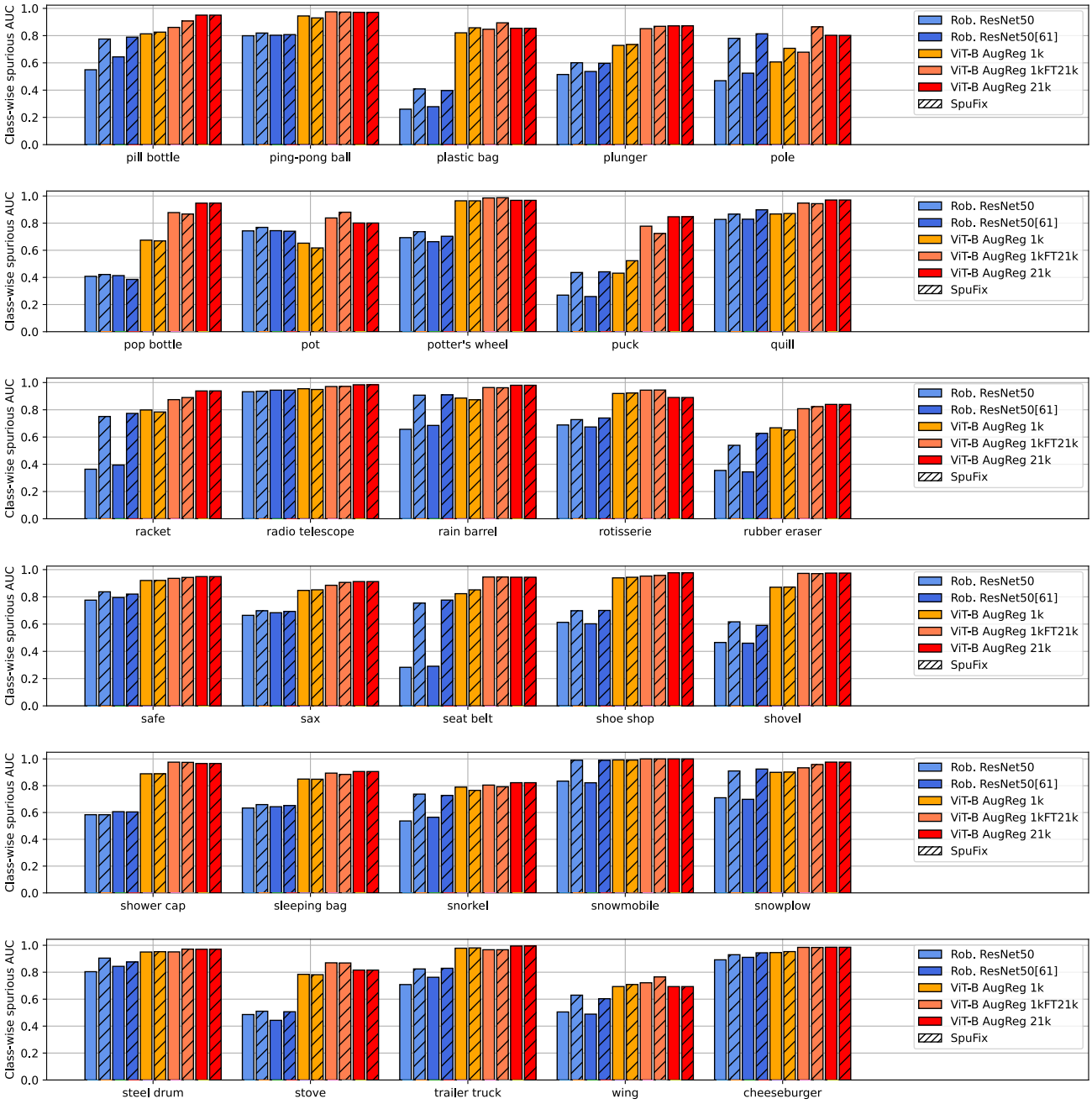

Figure 15: Continued from Fig. 13 for classes 71-100.

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.

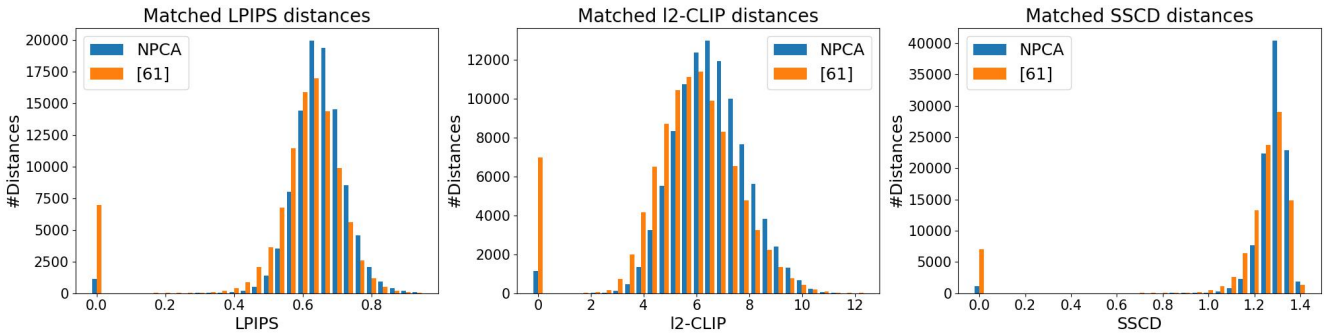

Figure 17: **Histograms of matched distances:** We consider the 5 maximally activating training images for each of the top-5 NPCA components resp. top-5 neurons of [61]. For each of these images (and each of the components/neurons) we find the best matching maximally activating training image of a different 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 l2-distance of clip embeddings [23], the SSCD distance used for image copy detection [47].

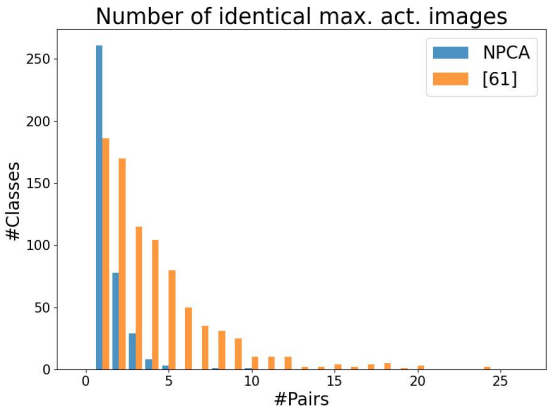

Figure 18: **Histogram of identical maximal activating training images for the top-5 NPCA components resp. the top-5 neurons of [61] for the robust model used by [61]** We observe that for NPCA only a few components have identical maximally activating training images and if it happens in the majority 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 activating 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.

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

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

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 μ_t is of the form

$$\begin{aligned} g_{\text{update}} &= C_c g_c - C_d g_d + C_a g_a, \\ \mu_t &= \mu_\theta(x_t, t) \\ &\quad + \Sigma_\theta(x_t, t) \left\|\mu_\theta(x_t, t)\right\|_2 g_{\text{update}}, \\ p(x_{t-1}|x_t, \hat{x}, k) &= \mathcal{N}(\mu_t, \Sigma_\theta(x_t, t)), \end{aligned}$$

where $g_c := n(\nabla_{x_t} \log p_{\text{robust},\psi}(k|f_{\text{dn}}(x_t, t)))$ is the normalized gradient of the adversarially robust classifier, $g_d := n(\nabla_{x_t} d(\hat{x}, f_{\text{dn}}(x_t, t)))$ - normalized gradient of the distance term. We add as additional guidance $g_a := n(\nabla_{x_t} \alpha_j^{(k)}(f_{\text{dn}}(x_t, t)))$ - 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 $\hat{x}_0 = f_{\text{dn}}(x, t)$ 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.

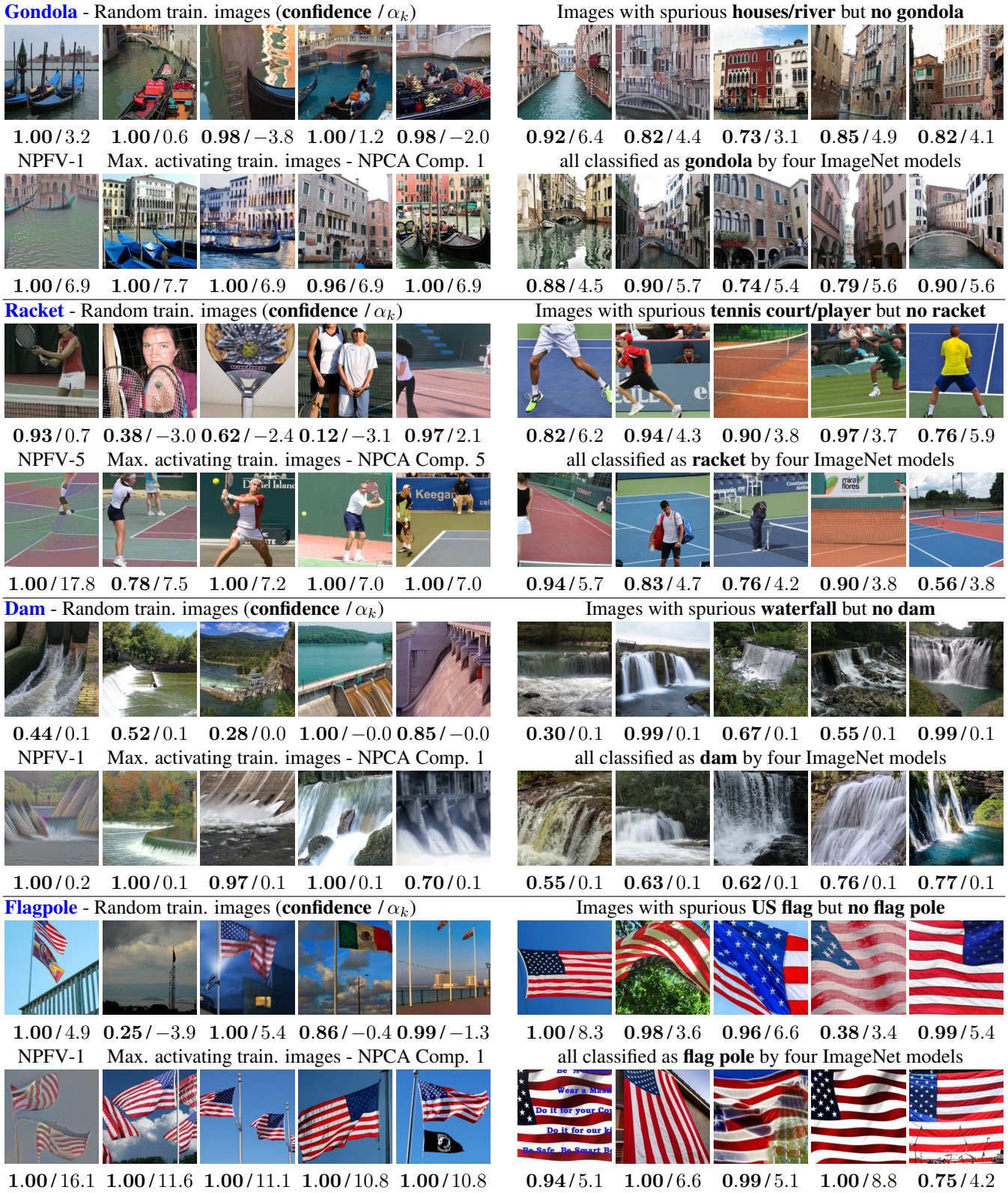

Figure 21: **Spurious features (ImageNet):** found by human labeling of our neural PCA components. For each class we show 5 random train. images (top left), the neural PCA Feature Visual. (NPFV) and 4 most activating train. images for the spurious feature component (bottom left). Right: four ImageNet 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.

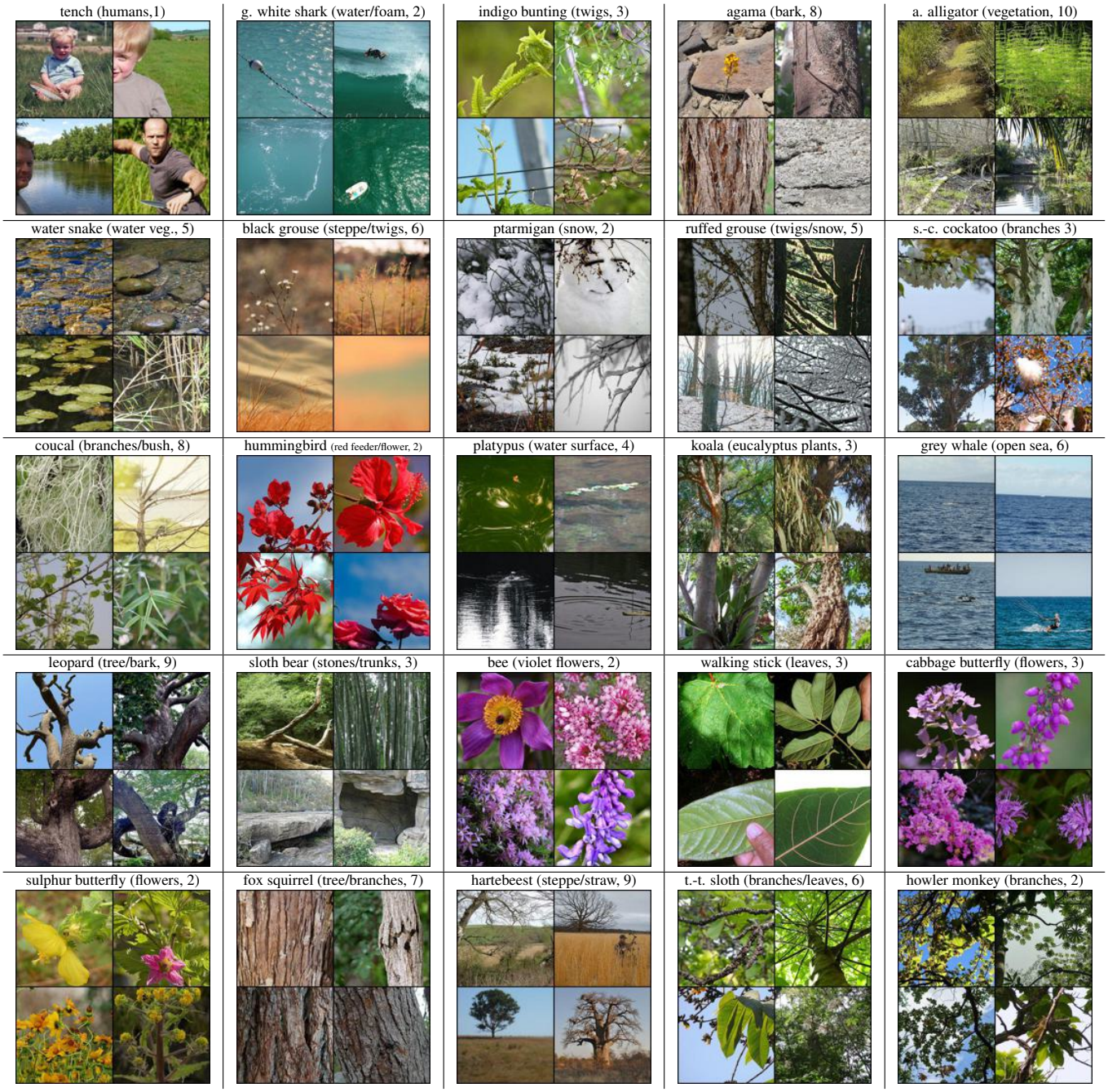

Figure 23: Random selection of 4 images for classes 1-25 of our "Spurious ImageNet" dataset with class label (spurious feature, NPCA component). Note that the labels of spurious features are coarse and thus overlap e.g. several are leaves/branches/fowers. We are not able to identify if these are special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

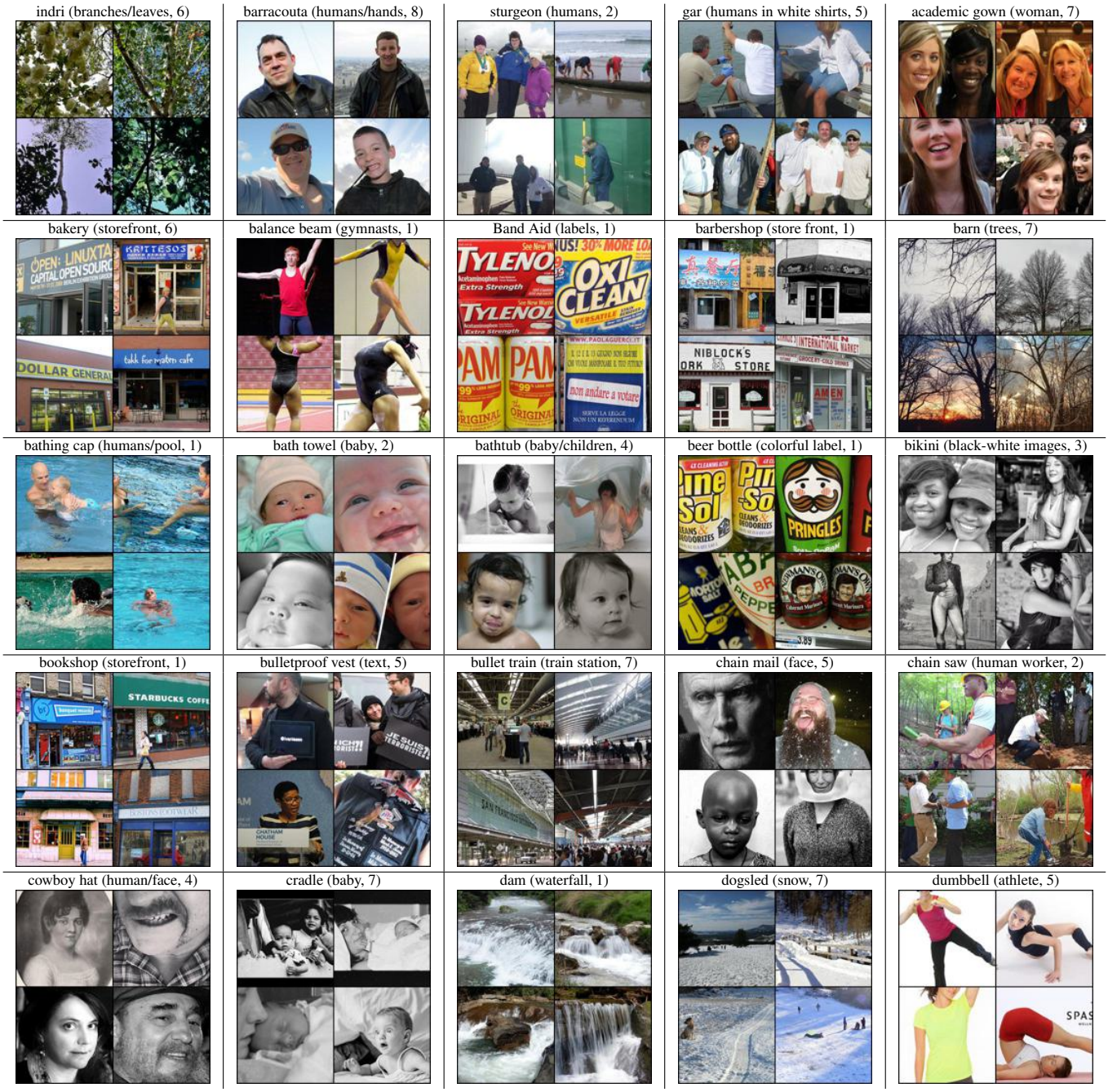

Figure 24: Random selection of 4 images for classes 26-50 of our "Spurious ImageNet" dataset with class label (spurious feature, NPCA component).

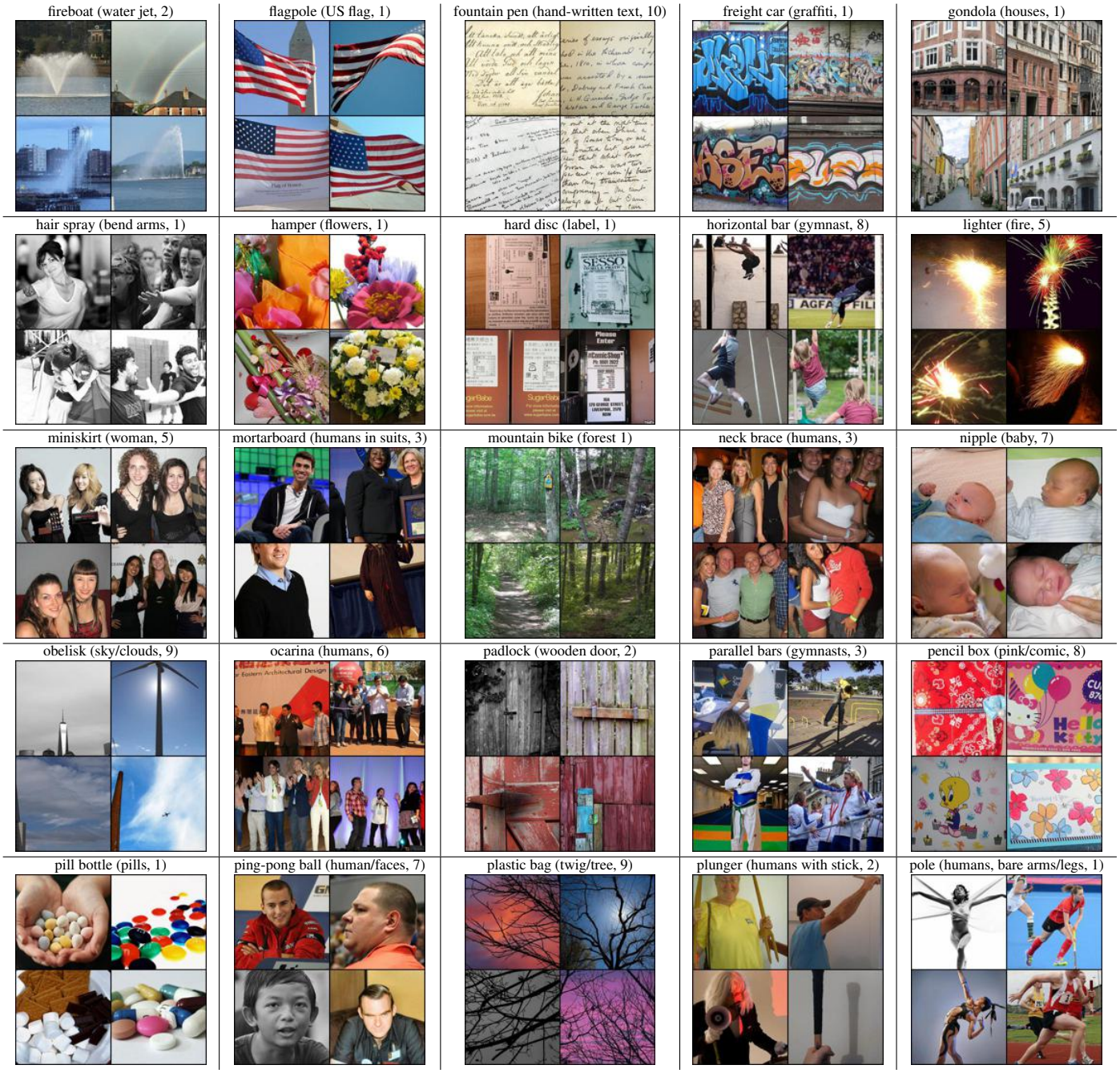

Figure 25: Random selection of four images for classes 51-75 of our "Spurious ImageNet" dataset with class label (spurious feature, NPCA component).

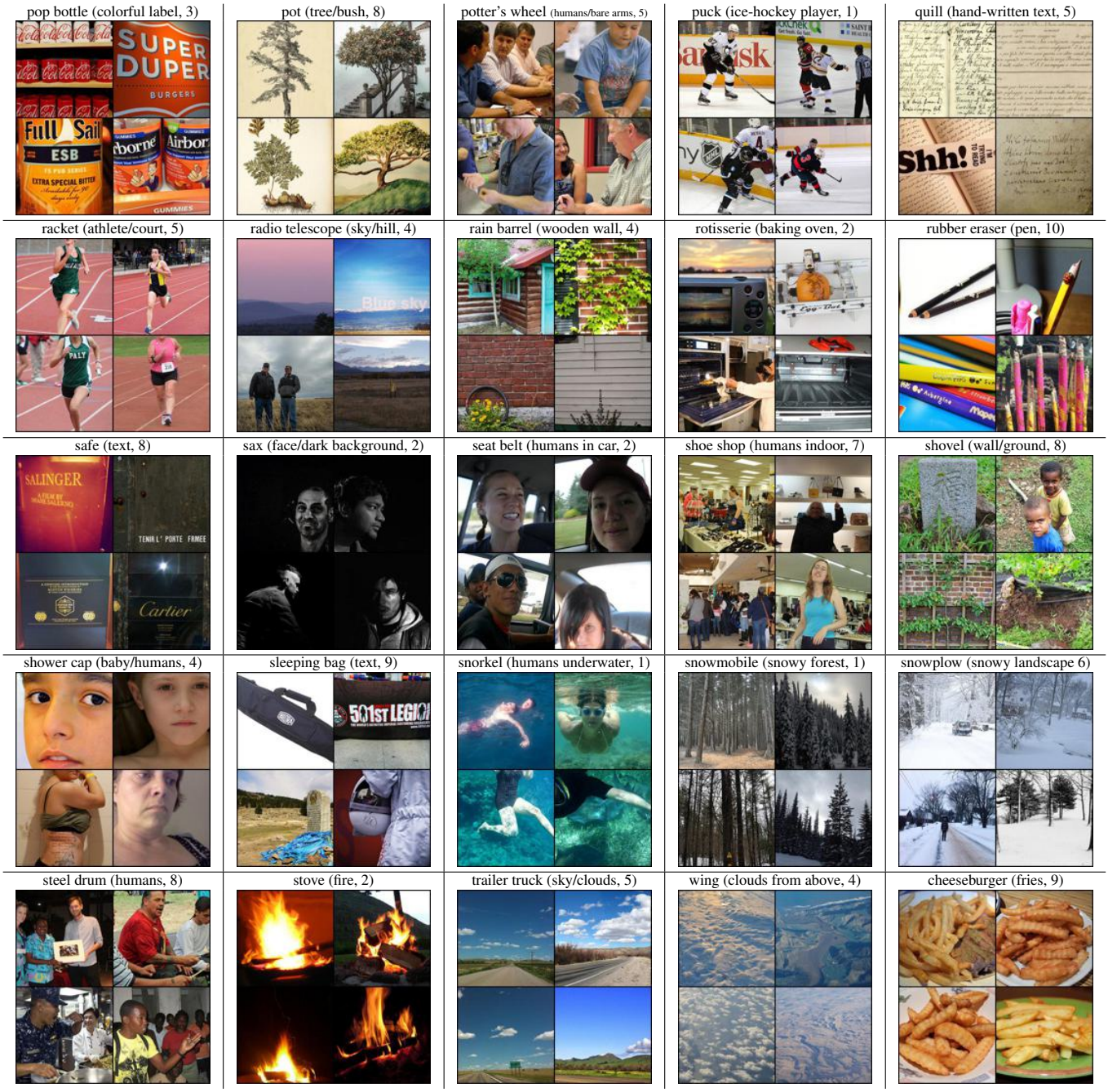

Figure 26: Random selection of 4 images for classes 76-100 of our "Spurious ImageNet" dataset with class label (spurious feature, NPCA component).

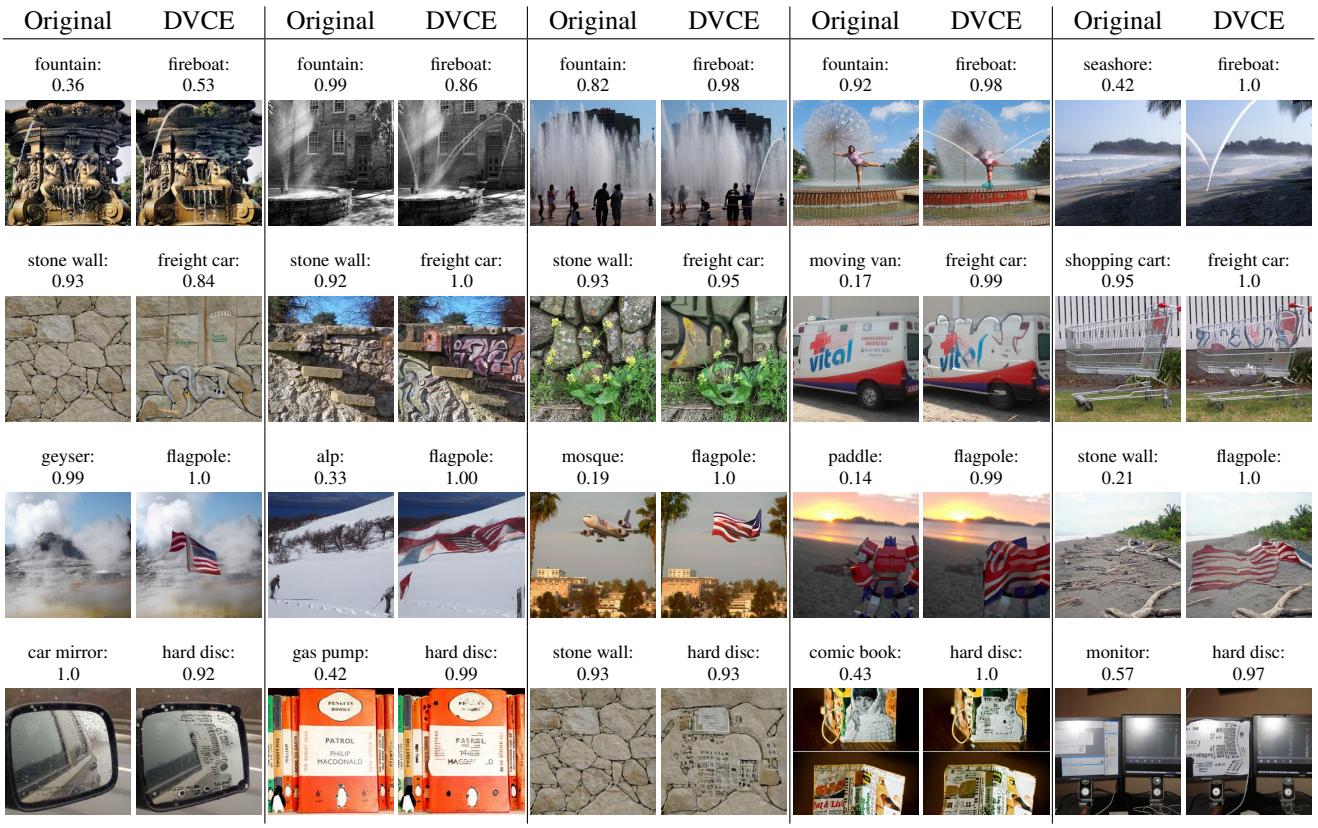

Figure 27: Adding spurious features automatically with an adaptation of DVCEs [5] changes the prediction of the classifier robust ResNet50. This happens, because, as has been shown qualitatively in Fig. 4 and quantitatively in Fig. 6, this classifier has learned to associate class "fireboat" with the 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.