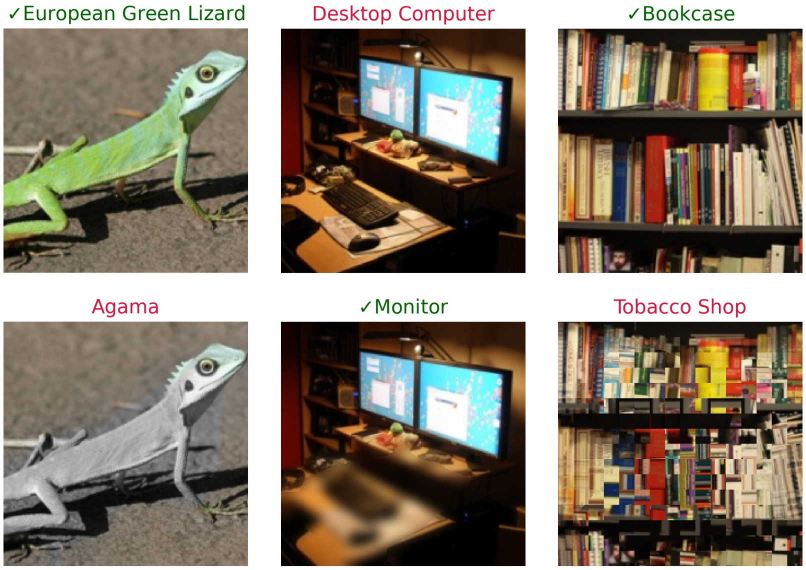
Appendix

A Impact and Limitations

We hope our work will allow for better understanding of how spurious feature reliance affects deep models, leading to improved handling of algorithmic fairness and model robustness, which are both related to the spurious correlation problem. We open source all code, and modify a key framework of our method to vastly increase its usability for people with fewer computational or data resources. We also create a website for easier viewing of our analysis, making our entire annotation process completely transparent. We obtain verbal approval[*] from our institution's IRB for our study, and pay workers above minimum wage. Key limitations are that our method does not detect all spurious features (only a subset), and uses a single model to detect and measure them, though we argue that the features we detect are very relevant and likely relied upon by all models; we elaborate on this point in Section I.3.

B Characterizing Feature Sensitivities

Using visualizations from our robust neural feature annotation, we obtain soft segmentations for features in input space that activate a specific neural feature. Namely, the Neural Activation Map for a neural feature can serve as a soft segmentation for the feature. Using a Mechanical Turk human study, we validate that neural activation maps for images in the top 20^{th} percentile within its class for activation of the neural feature segment the same input pattern (see Appendix I.2.2).



(a) Gray (Color) (b) Blur (Texture) (c) Patch Rotate (Shape)

Figure 10: Examples of features that are particularly sensitive to color, texture, and shape corruptions. Original images (**top**) and corrupted images (**bottom**) shown, along with model predictions. In one case, corrupting the spurious feature of *keyboard on desk* for class *monitor* actually improves prediction.

We then use targeted corruptions to gain insight on the information within an input region that models rely on most. The core premise of our argument is that degradation in model performance due to corruption of some information can be a proxy of the importance of the corrupted information. Indeed, similar arguments were made in [46, 29], where adding Gaussian noise to specific regions was employed to assess core/spurious or foreground/background sensitivity respectively. However, in our

---

[*] We were informed that formal approval is not needed because we do not obtain information *about* our crowd worker, and thus our study does not constitute human-subject research.

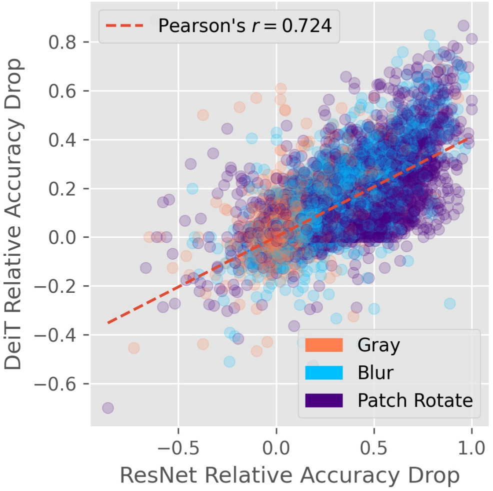


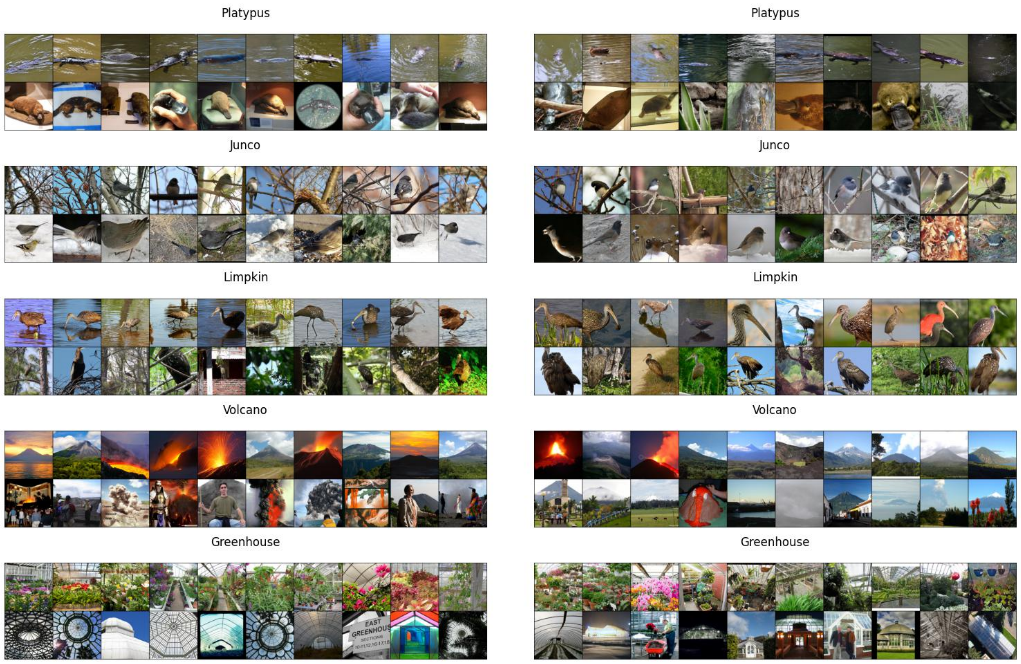
Figure 11: Sensitivities of each class-feature pair to specific corruptions for ResNet-50 and DeiT-Small. The two distinct models rely on features in the *same* way.

analysis, we use carefully chosen corruptions so to remove one aspect of primitive information (i.e. color, shape, or texture) without disrupting the others. Namely, our corruptions are graying out image regions, blurring out image regions, or patching an image region and randomly rotating each patch by a multiple of 90 degrees. Graying out an image removes color, blurring out destroys local (i.e. textural) details, and random patch rotation breaks longer edges, hence disrupting shape information.

We apply these targeted corruptions to segmented feature regions for all 5000 annotated class-feature pairs. Specifically, we focus on the 65 images per class where a feature is activated most highly (i.e. roughly top 5^{th} percentile), since the feature is most prominent in these images. Also, for spurious features, we include a filtering step to avoid corrupting core regions. We do this by removing any overlap of the spurious feature segmentation with the consolidated core mask [46], which is a pixel-wise maximum of soft segmentations for any relevant core features. We track model accuracy on samples with and without each corruption applied, using change in accuracy as a measure of sensitivity.

Figure 10 shows examples of features that are the most sensitive (i.e. highest change in accuracy) to each of the three corruption types, visualizing the corruption and its affect on model prediction. In the left panel, we see that the color information in the core feature of *lizard body* is crucial for successful prediction of the class *European Green Lizard* (matching intuition). In the middle panel, we see a case where the spurious feature of *keyboard on desk* hurts classification. Namely, blurring out the keyboard and desk leads to more accurate prediction of the *desktop computer*. We note that while the shape and color information is retained, it appears that the local details in the texture of the keyboard and desk contribute more to confusing the model and leading to misclassification. Lastly, for the spurious feature of *books* on a *bookcase*, we observe that the shape of the books is crucial, as rotating patches leads to an incorrect prediction of *tobacco shop*.

We conduct this analysis using two pretrained models of roughly equal size: ResNet50 and small DeiT. Figure 11 shows the relative accuracy drops incurred by performing each of the three corruptions for all 5000 annotated class-feature pairs on both models. We find that the two models have strongly correlated sensitivities ($r = 0.724$). The least correlated sensitivities appear to be for the patch-rotate corruption, where the transformer has nearly zero accuracy drop for many cases where the ResNet experiences greater accuracy drop. This is most likely due to the unique robustness of vision transformers to various patch transformations like permutation, rotation, and ablation [34, 36]. Despite this, the strong correlation between feature sensitivities for two diverse models suggests that model behavior is determined much more by the data it operates over than the specific decisions made during training. In other words, the features that models rely on and the ways in which they rely on them may be far more a function of data than a function of the model. If this conjecture holds, it would warrant greater inspection of the role of data in various open problems in modern deep learning (robust generalization, efficient learning, etc).



(a) Training Split

(b) Validation Split

Figure 12: Images with highest and lowest spuriosity for five randomly selected classes (top/bottom row in each chunk corresponds to images with highest/lowest spuriosity respectively). The spurious features depicted are *lake water, branches, water, sky with clouds or lava*, and *plants*. The spurious features are easier to see in the training split than in the validation split, because there are roughly $25\times$ more samples to choose from. Nonetheless, the validation images still seem to be organized so that the top-10 images contain the spurious feature much more than the bottom-10, indicating that our spuriosity ranks generalize to the validation split.

C More Examples of Spuriosity Rankings

To further validate our spuriosity rankings, we now present more examples of images ranked by spuriosity. While this validation is qualitative, we argue that it is sufficient since we are attempting to proxy a fundamentally qualitative notion. Namely, in figure 12, we present images organized by spuriosity ranking for five randomly selected classes. Here, we show images both ranked in the training and validation split to demonstrate that even though trends are clearer when ranking training images (because there are $\sim 25\times$ more of them to choose from), the underlying semantic concepts by which images are ranked still generalize to the validation split.

D Full Pretrained Model Evaluation Results

We now provide greater detail for the pretrained model evaluation presented in Section 4.2. We evaluate 89 models spanning diverse architectures and training procedures. Namely, the models we consider fall into the following categories, with category nicknames in parenthesis: supervised vision transformers (Sup ViTs), self-supervised vision transformers (SS ViTS), CLIP vision transformers (CLIP ViTs, both Zero-Shot and Finetuned), CLIP ResNets (CLIP RNs), self- or semi-supervised ResNets (SS ResNets), supervised ResNets (Sup ResNets), mixer and ResMLP models (MLP-Based), various other convolutional networks (Other CNNs), and adversarially trained ResNets (Robust ResNets). The vast of majority of pretrained weights are obtained from the timm library [51], while others come directly from their original respective repositories [7, 9, 37, 43]. Figure 13 shows effective robustness for each model, as well as the average effective robustness per category.

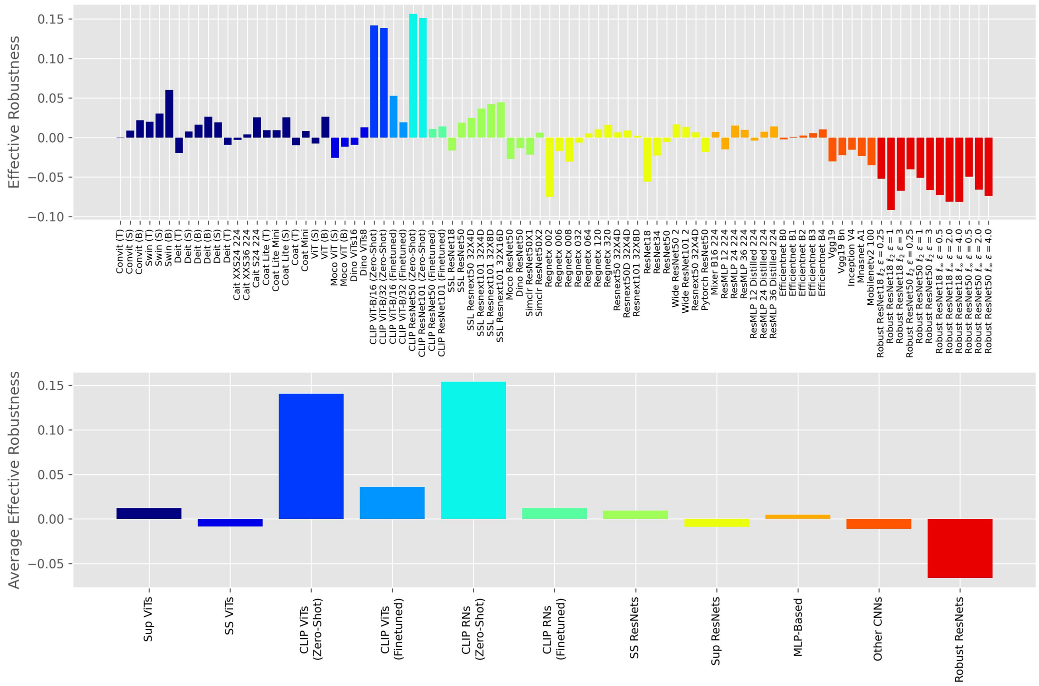


Figure 13: Effective robustness for all 89 models considered (**top**), along with effective robustness averaged over model category (**bottom**).

We measure effective robustness by first fitting a line to predict accuracy on the 10 validation images per class with the *least* spuriosity as a function of accuracy on the 10 validation images per class with the *most* spuriosity. Let acc_{top} and acc_{bot} denote accuracy on images with the most and least spuriosity respectively (i.e. top and bottom with respect to spuriosity rankings). We refer to the line of best fit as $\beta : \mathbb{R} \to \mathbb{R}$. Effective robustness for a model with accuracies on top and bottom spuriosity ranked images $\text{acc}_{\text{top}}, \text{acc}_{\text{bot}}$ respectively is simply $\text{acc}_{\text{bot}} - \beta(\text{acc}_{\text{top}})$. In other words, we measure how high above the line of best fit a model's marker falls in figure 6 (left).

We note that this differs slightly from the originally presented notion of effective robustness in [27], where a log-linear fit is performed. Moreover, instead of predicting out-of-distribution accuracy based on in-distribution accuracy, we compare accuracy on two carefully chosen subsets of in-distribution data. We argue that we capture subsets where subpopulation shift has occurred, specifically with respect to common spurious cues for each class.

Our findings show that some models are more effectively robust than others. Namely, zero-shot CLIP models have much higher accuracy on images with lowest spuriosity than their performance on the images with highest spuriosity would predict. After finetuning a linear head using ImageNet on top of the fixed CLIP image encoder, we see effective robustness drops significantly. We note that both these results were observed using independent distributional robustness measures in [53]. At the other extreme, adversarially trained ResNets have the lowest effective robustness, a result carefully studied in [28].

Aside from these two extremes, most trends are muted. We believe the stronger signal exists in per-class spurious gaps, as shown in figure 6 (right), suggesting that while models generally behave similarly on the same inputs, their behavior (i.e. with respect to spurious feature dependence) varies dramatically across inputs. Indeed, we find that per-class spurious gaps correlate strongly between pairs of models in our evaluation suite. Figure 14 visualizes these correlations, with an average correlation of $r = 0.69$ being observed. We thus recommend closer attention to be paid to the individual data classes and potential spurious features *per class* a model is to be deployed over.

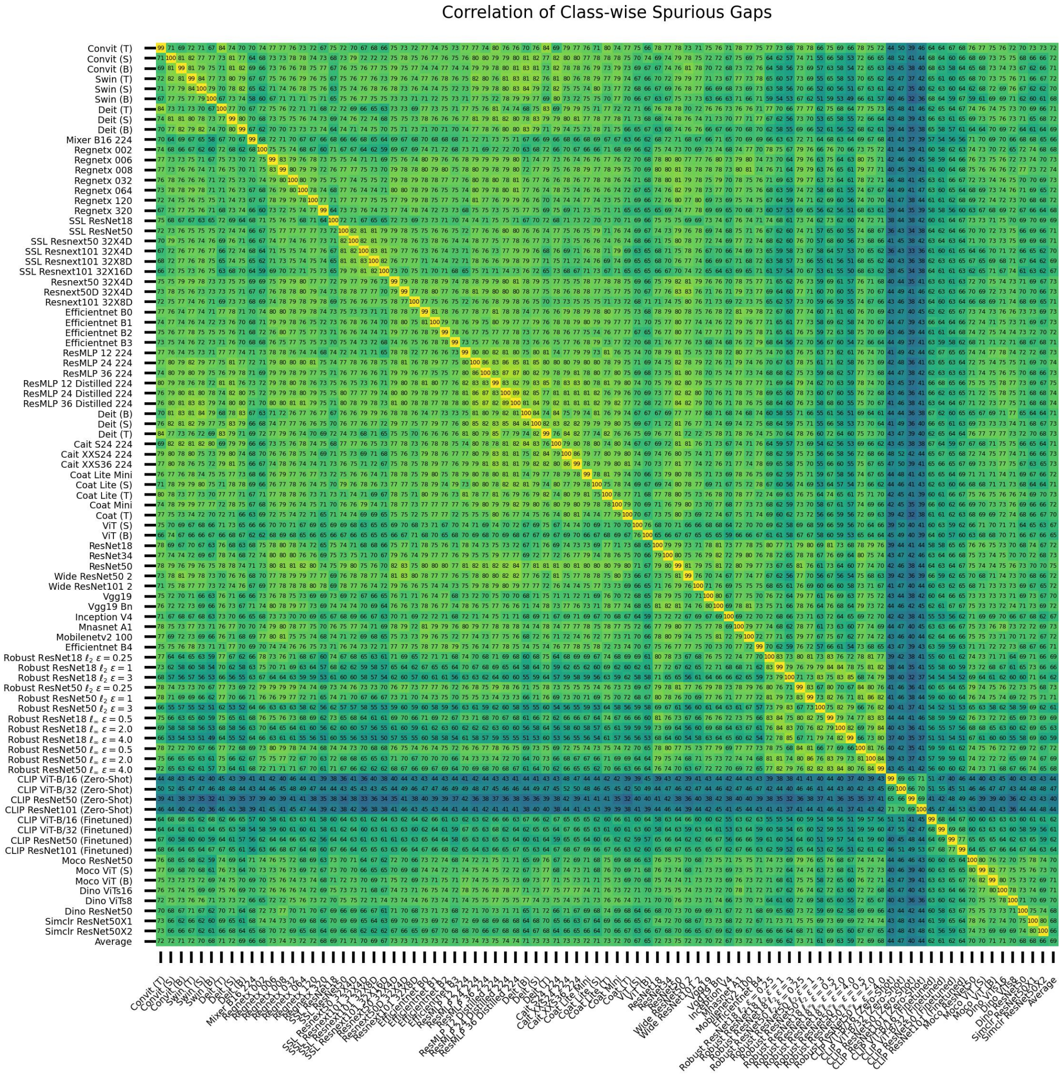
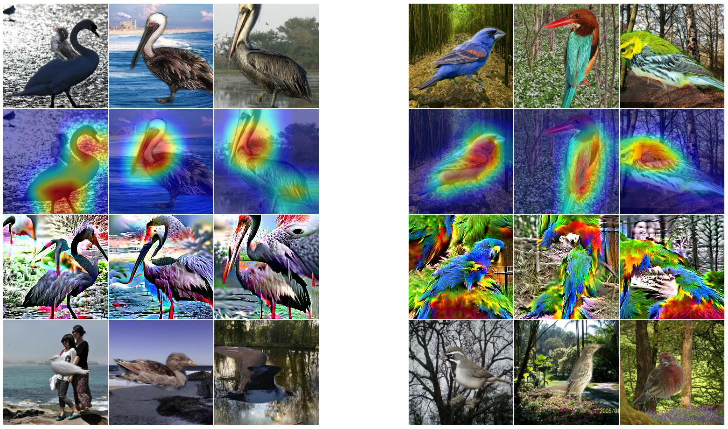


Figure 14: Correlation of per-class spurious gaps for all pairs of models. Numbers shown are Pearson's r correlation coefficients times 100. On average, a pair of diverse models have strongly correlated class-wise spurious gaps, with $r = 0.69$. Notably, zero-shot CLIP models have the worst correlation with other models, suggesting that their perception is fundamentally different.

E Details on Feature Discovery Beyond ImageNet

E.1 General Training Details

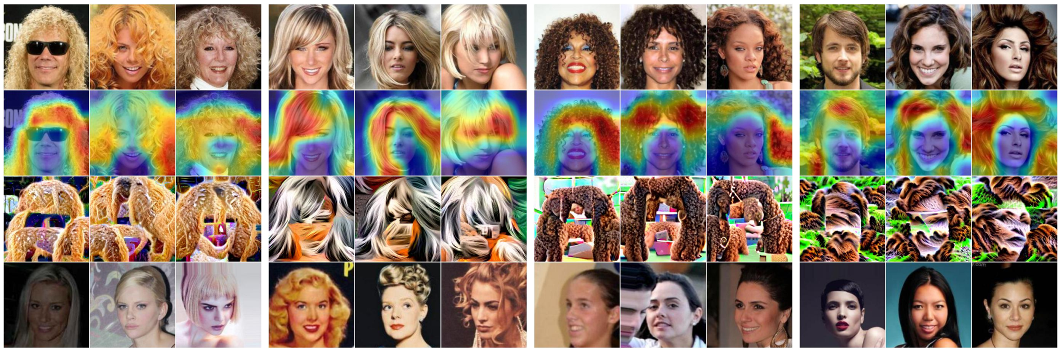
Recall that we introduce a lightweight extension of spurious feature/concept discovery without requiring adversarial training by simply fitting a linear layer on new data over a fixed adversarially trained feature encoder (Section 3.2). Specifically, in all cases, we use a ResNet50 feature encoder (i.e. all layers except for the final linear classification head) adversarially trained for ImageNet classification with projected gradient descent using an ℓ_2 norm attack of budget $\epsilon = 3.0$, downloaded from [43]. Note that this feature encoder comes from the same network used to discover core and spurious features in ImageNet. However, we train new linear heads over fixed features for four tasks over three datasets: Waterbirds [42], Celeb-A [25], and UTKFace [60]. In all cases, we use an Adam optimizer with learning rate of 0.1 and weight decay of 0.003. We train for 20 epochs. While the model we use may not be as accurate as a model trained from scratch, they still achieve relatively high accuracies given the ease of their training. Notably, for Celeb-A hair task (2-way), UTKFace gender task (2-way), and UTKFace Race task (5-way), our linear layer fitting on robust neural features yields



(a) Waterbird core feature. Focus is on elongated and curved necks.

(b) Landbird core feature. Focus is on feathers with bright colors.

Figure 15: Core features discovered for the Waterbirds waterbird vs. landbird classification task. We identify features that occur in specific subpopulations of the broader classes. Low activating images reveal subpopulations where the feature is absent. The rows show (from top to bottom): highly activating images, corresponding heatmaps and feature attacks, and lowly activating images.



(a) Blond curly hair.

(b) Blond straight hair.

(c) Brown curly hair.

(d) Brown wavy hair.

Figure 16: Core features discovered for the Celeb-A blond vs. brown hair classification task. We identify features corresponding to different types of hair within the same broader class of hair color. The rows show (from top to bottom): highly activating images, corresponding heatmaps and feature attacks, and lowly activating images. The captions reflect the focus of the robust neural feature.

accuracies of $95.3\%, 86.5\%$, and 66.7% respectively. Acccuracy on the Waterbirds validation set is lower (77.6%), though this is typical for Waterbirds, as the validation shift features a significant subpopulation shift (by design, due to breaking the background correlation). More importantly, in all cases, important neural features are observed to correspond to data patterns (i.e. human concepts) salient to the task at hand, including spurious ones.

E.2 Additional Discovered Features for Waterbirds and Celeb-A

We now present additional features identified via our lightweight extension of the feature discovery framework of [46]. Specifically, we show core features, which are relied upon more frequently than spurious features, which we show in Figure 4. We observe robust neural features to respond to specific patterns shared among subpopulations within the broader classes. Surprisingly, these robust neural features were not at all trained on images from the downstream tasks. Yet, the patterns they respond to are semantically meaningful for the downstream task. We attribute this to the diversity in ImageNet, which leads to learning many general visual patterns that are manifested in more specific contexts for a vast number of downstream tasks.

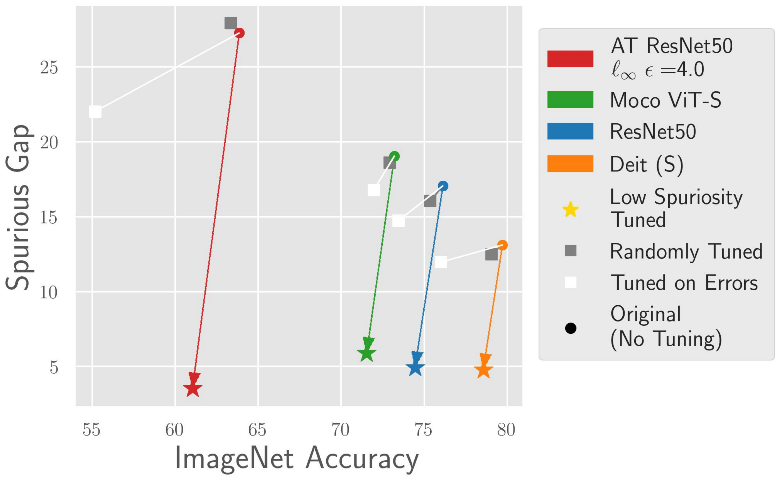


Figure 17: In addition to tuning on low-spuriosity images and on random samples, we include results from tuning on misclassified samples in white. Error tuning minimally closes spurious gaps, and actually leads to substantial drops in overall validation accuracy.

E.3 UTKFace Experiments

UTKFace [60] is an attributed dataset originally intended for the task of synthetically aging a face image using conditional generative models. The dataset consists of over $20,000$ images with age, ethnicity, and gender annotations. We consider the task of gender and age classification. We use 90% of the data for training, and hold out the remaining 10%. We inspect the 15 most important features for each class per task. For all classes and tasks, we are able to identify at least on spurious feature with our method. Notably, these tasks were not designed to contain spurious features, like the other two benchmarks analyzed. This goes to show that spurious correlations may arise in any setting, and our method can reveal such spurious features with minimal computation. Having discovered robust neural features that detect spurious concepts, one can sort data by spuriosity to measure and mitigate the resultant bias.

F Alternate Baseline: Error Tuning Does Not Close Spurious Gaps

A common approach within recent spurious correlation literature is to automatically select samples to conduct additional training on by simply inspecting whether they were classified correctly originally. For example, Liu et al. [24] propose to 'just train twice', upweighting error samples in the second round of training; Nam et al. [32] 'learn from failure' by training one classifier that is intentionally biased, and using the hard (i.e. misclassified) samples from the biased classifier to obtain an unbiased classifier; Zhang et al. [59] use error samples as hard positives in an additional round of contrastive training. Inspired by these approaches, we include an additional baseline to the experiment from section 4.3 where we tune existing classification heads on samples originally misclassified. Specifically, we train on misclassified samples for all classes. All three methods train on 100 samples per class. When the number of errors in a class is less than 100, we add randomly selected correctly classified class instances. When the number of errors is more than 100, we randomly select 100 of them. Results shown are averaged over three trials.

We observe that while error tuning closes spurious gaps more than tuning on random samples, the reduction in spurious gaps is far less than that from low spuriosity tuning. More importantly, overall validation accuracy drops substantial when tuning on errors. We conjecture this occurs because errors can be caused by many things aside from the absence of spurious cues, such as label noise. Thus, tuning on misclassified samples could lead to overfitting on unreliable data. In contrast, low spuriosity images are designed to only differ from typical samples in that they lack particular cues that have already been deemed spurious by a human. Arguably, low spuriosity images can at times offer more reliable learning signal, as they do not contain distracting shortcuts (e.g. the low spuriosity lighters in figure 1 are far easier to see than the high spuriosity ones). Also, while the error-centric approaches were shown to be effective on existing spurious correlation benchmarks, we hypothesize that the biases in those benchmarks are overly simplistic: they typically only include one spurious correlation and at most a handful of classes. Therefore, these methods may struggle in more realistic settings, as demonstrated by their ineffectiveness on this ImageNet scale experiment.

Thus, we claim that low spuriosity tuning offers unique advantages compared to prior work. While most existing methods focus on altering the training algorithm, our method acknowledges that data may have a larger (and often overlooked) role in determining the biases a model absorbs, and focuses on finding the right data to tune. The experiments in this section show that compared to another line of data-centric approaches, our low spuriosity tuning may be more effective, in both retaining overall accuracy and reducing biases.

G On Automating Spuriosity Rankings

We believe that human involvement is a strength of our framework, as it increases transparency. Namely, the human in the loop is given a concise inside look on the cues a model trained on the given data is likely to rely upon. Moreover, the human is given agency to decide which cues model performance should be invariant to. Also, the level of human involvement is relatively low. Given the complexity of the task of sorting thousands of images within a class, only requiring a human to inspect a handful of images is relatively efficient. With an appropriate UI, we can confirm that this takes no longer than about a 30-40 seconds per class.

However, in cases where minimizing cost is preferred, Spuriosity Rankings can be automated. One way to do so is to automate the annotation of neural features as core or spurious. Specifically, one could automatically segment the class object with open-vocabulary segmentation models [21], and then compute the amount of saliency placed on the object by the neural activation map. If most of the salient pixels for a neural feature lie outside of the image region containing the class object, one could automatically flag such a feature as spurious.

Another way to automate Spuriosity Rankings is to leverage vision-language models (VLMs) like CLIP [37]. With VLMs, we can compute the similarity of an image to any concept, encoded directly from text. After sorting images by similarity to spurious concepts, we can inspect the Spurious gap to measure bias, and train on low spuriosity images to mitigate bias. Thus, one must simply enumerate (in the form of text) potential relevant spurious cues; since encoding the cue and sorting along it is free, one can discard the cues which do not result in an accuracy gap between the most and least similar images, as the model of interest apparently would not be biased to the presence of that cue. We note that while VLM training is expensive, a pre-trained model would suffice, and moreover, recent work shows that VLM abilities can be extended to arbitrary vision-only models cheaply as well [31].

Note that the underlying mechanism of Spuriosity Rankings is preserved in both of these automateed variants: we utilize the representation space of an interpretable model to quantify the presence of relevant cues, and rank images by spuriosity (proxied by similarity/activation of concept directions in representation space) to enable interpretation of spurious cues in context and measure model bias caused by them.

H Details on Core-Cropping

To perform core-cropping, we first obtain a soft segmentation of core regions, done by averaging the neural activation maps for core neural features. Next, we threshold the averaged mask to only retain highly activated pixels. We use a threshold of 0.9. Then, we obtain a bounding box to localize a rectangular region encompassing the pixels that most highly activate the core region. Since we use a high threshold, we expand our bounding box by 20% along both height and width dimensions. Finally, we convert our rectangular region to a square one by simply extending the shorter side to match the length of the larger side. We note that this method is only as effective as the core soft segmentation. Note that core neural activation maps are generally reliable for images that reasonably activate the core features. Namely, we validate the quality of these segmentations for images in the top 20^{th} percentile of activation for a class (see Appendix I.2.2). However, for the remaining 80% of images, these soft segmentations may not be perfectly reliable. Nonetheless, as segmentation models improve, core-cropping may also improve, as we can simply replace core soft segmentations obtained via neural activation maps with actual segmentations, conditioned on the class object. Thus, flagging potentially mislabeled samples

I Robust Neural Feature Annotation Details

Our work uses the feature discovery framework introduced in [46], also known as Salient ImageNet. We now i. introduce a taxonomy of relevant terms, ii. review the framework, iii. discuss its merits and potential limitations, and iv. detail our expansion/modifications to it.

I.1 Taxonomy of Relevant Terms

Below we enumerate and explain some terms we refer to throughout the paper.

- **Core** feature or cue: a visual pattern that is essential to the class object; i.e., the pattern is a part of the class object, like fur for an otter.
- **Spurious** feature or cue: a visual pattern that is not essential to the class object; i.e., the presence of the spurious cue is not required for the image to belong to the class, and the spurious cue can exist in images from other classes.
- **Neural Feature**: node in the penultimate layer of a classifier parameterized by a deep neural network. Importantly, each logit of the classifier is a linear function of neural features.
- **Robust neural feature**: neural feature from an adversarially trained network. We use an ℓ_2 PGD-trained [26] network with $\epsilon = 3$.
- **Neural activation map**: an extension of a class activation map [61] for neural features, used to highlight the region of an image responsible for activating a neural feature.
- **Feature attack**: a visualization technique to amplify the visual cues present in an image that activate a neural feature.

I.2 Salient ImageNet Framework

The steps of the Salient ImageNet Framework are:

1. Adversarially train a model for your classification task.
2. Identify important neural features per-class based on feature importance, which is simply the product of the average feature activation and the weight of linear head connecting feature to class logit. Feature importance explicitly computes the average contribution of a feature to a class logit for samples in the class.
3. Select the top-k (i.e. $k = 5$) activating instances of a class for a selected feature and additionally generate heatmaps (via overlaying the Neural Activation Map on the original image) and feature attacks (via adversarially attacking the original image using gradients from the adversarially robust model so to maximize activation on the feature).
4. Have humans inspect the $3 \times 5 = 15$ visualizations to determine if the focus of the neural feature is on the main object (i.e. pertaining to the class and hence *core*) or on the background or a separate object (hence making the feature *spurious*).

We elaborate on the feature discovery and annotation procedure in subsection I.2.1, including an example of the MTurk form. In [46], the annotated robust neural features were primarily used to softly segment image regions containing core and spurious features at scale. Thus, in addition to the Mechanical Turk study performed to annotate neural features, a second study was conducted to confirm that neural activation maps for annotated features softly segment the same input features across the top 5^{th} percentile of samples within the relevant class. With this confirmation, they computed neural activation maps for all of these images and used these as feature segmentation maps. To assess reliance on core and spurious features, they would inspect drop in accuracy due to adding small amounts of Gaussian noise to these regions. We repeat a modified and expanded version of this second study, so to validate a larger number of feature segmentations. Details on this human study are in subsection I.2.2.

I.2.1 Mechanical Turk study for discovering spurious features

The design for the Mechanical Turk study is shown in Figure 18. The left panel visualizing the neuron is shown in Figure 18a. The right panel describing the object class is shown in Figure 18b. The questionnaire is shown in Figure 18c. We ask the workers to determine whether they think the visual attribute (given on the left) is a part of the main object (given on the right), some separate object or the background of the main object. We also ask the workers to provide reasons for their answers and rate their confidence on a likert scale from 1 to 5. The visualizations for which majority of workers selected either background or separate object as the answer were deemed to be spurious. Workers were paid \$0.1 per HIT, with an average salary of \$8 per hour. In total, we had 137 unique workers, each completing 140.15 tasks (on average).

We conduct this study for the five neural features with highest contribution to each of the 768 ImageNet classes not analyzed in [46], resulting in 3840 newly annotated class-feature dependencies, including 468 spurious ones ($3.3\times$ and $2.9\times$ increases respectively).

I.2.2 Mechanical Turk study for validating heatmaps

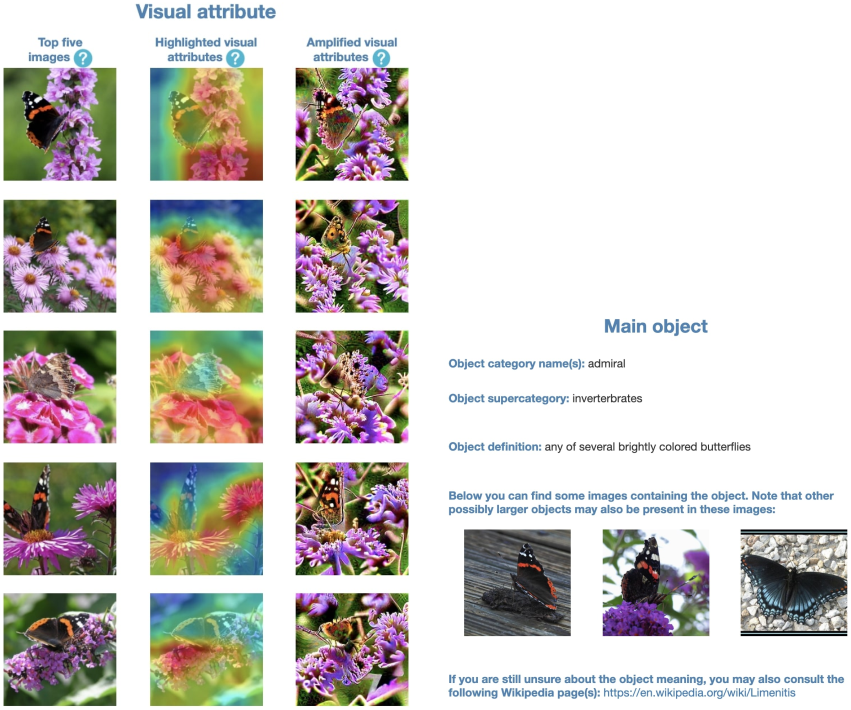
We now detail the second MTurk study we carry out to validate the feature soft segmentations generated by Neural Activation Maps of annotated robust neural features. For each annotated class-feature dependency, we first obtain the training images from the class who's activation on the feature is within the top 20^{th} percentile for the class. This results in a set of 260 (20% of 1300 training images per class) images per class-feature pair, as opposed to only 65 in [46]. From this set of 260 images, we selected 5 images with the lowest activations on the neural feature and randomly selected 10 images from the remaining set (excluding the already selected images). We show the workers three panels: images and heatmaps with (i) the highest 5 activations, (ii) the next 5 highest activations, and (ii) the lowest 5 activations. For each panel, workers were asked to determine if the highlighted attribute looked different from at least 3 other heatmaps in the *same panel*. Next, they were asked to determine if the heatmaps in the 3 *different panels* focused on the same visual attribute, different attributes or if the visualization in any of the panels was unclear. Thus, we validate that both within each panel and across panels, the focus of the neural feature is consistent.

The design for the Mechanical Turk study is shown in Figure 19. The three panels showing heatmaps for different images from a class are shown in Figure 19a. The questionnaire is shown in Figure 19b. For each heatmap, workers were asked to determine if the highlighted attribute looked different from at least 3 other heatmaps in the same panel. We also ask the workers to determine whether they think the focus of the heatmap is on the same object (in the three panels), different objects or whether they think the visualization in any of the panels is unclear. Same as in the previous study (Section I.2.1), we ask the workers to provide reasons for their answers and rate their confidence on a likert scale from 1 to 5. The visualizations for which at least 4 workers selected same as the answer and for which at least 4 workers did not select "different" as the answer for all 15 heatmaps were deemed to be validated i.e for this subset of 260 images, we assume that the robust neural feature's focus is on the same visual attribute. Showing three panels as opposed to two is novel to our work and increases the standard for validation.

For all (class, feature) pairs ($1000 \times 5 = 5000$ total), we obtained answers from 5 workers each. We successfully validate 4763 pairs (i.e. 95.26%), demonstrating the effectiveness of using neural feature supervision to automatically (softly) segment an input pattern in a large number of images. We utilize these feature segmentations in Appendix B.

I.3 On Feature Discovery Using Only One Model

The key idea of our framework (expanded from [46]) was to use neural features of an adversarially trained network as automatic 'concept' detectors, and further to use their activation maps as soft segmentations. One may argue then that the features discovered are biased to the underlying model used. However, we argue that the primary patterns that models rely on are generally the same. Note that the existence of patterns predictive of a class is determined by the data distribution and not the model. While models may learn to rely on different patterns and may also vary the degree to which they rely on the same patterns, we believe that the most predictive patterns will be detected and relied upon by nearly all models. In our experiments, we find that the class-wise spurious gaps are highly correlated for any two pairs of models, with average correlation of $r^2 = 0.69$ (see figure 14). We also observe



Main object

Object category name(s): admiral

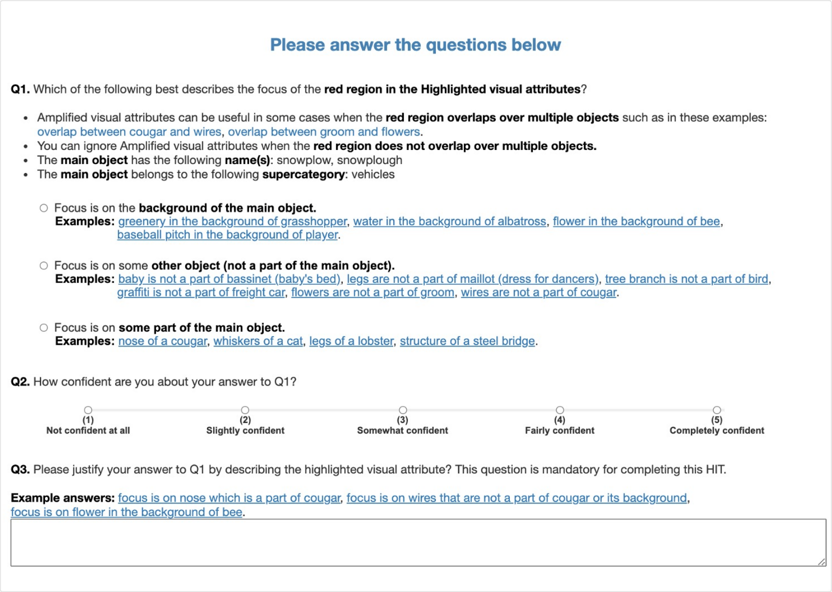
Object supercategory: inverterbrates

Object definition: any of several brightly colored butterflies

Below you can find some images containing the object. Note that other possibly larger objects may also be present in these images:

If you are still unsure about the object meaning, you may also consult the following Wikipedia page(s): https://en.wikipedia.org/wiki/Limenitis

(a) Visual attribute (b) Main object



Please answer the questions below

Q1. Which of the following best describes the focus of the **red region in the Highlighted visual attributes**?

- Amplified visual attributes can be useful in some cases when the **red region overlaps over multiple objects** such as in these examples: overlap between cougar and wires, overlap between groom and flowers.
- You can ignore Amplified visual attributes when the **red region does not overlap over multiple objects.**
- The **main object** has the following **name(s)**: snowplow, snowplough
- The **main object** belongs to the following **supercategory**: vehicles

○ Focus is on the **background of the main object.**
Examples: greenery in the background of grasshopper, water in the background of albatross, flower in the background of bee, baseball pitch in the background of player.

○ Focus is on some **other object (not a part of the main object).**
Examples: baby is not a part of bassinet (baby's bed), legs are not a part of maillot (dress for dancers), tree branch is not a part of bird, graffiti is not a part of freight car, flowers are not a part of groom, wires are not a part of cougar.

○ Focus is on **some part of the main object.**
Examples: nose of a cougar, whiskers of a cat, legs of a lobster, structure of a steel bridge.

Q2. How confident are you about your answer to Q1?

○ (1) Not confident at all
○ (2) Slightly confident
○ (3) Somewhat confident
○ (4) Fairly confident
○ (5) Completely confident

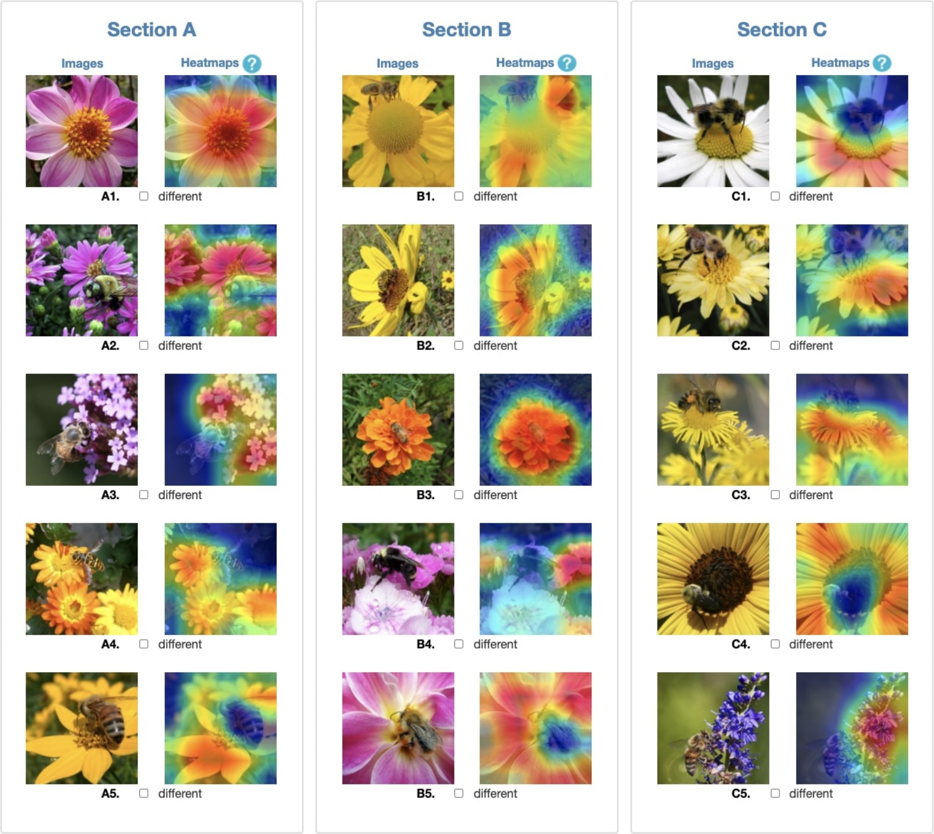
Q3. Please justify your answer to Q1 by describing the highlighted visual attribute? This question is mandatory for completing this HIT.

Example answers: focus is on nose which is a part of cougar, focus is on wires that are not a part of cougar or its background, focus is on flower in the background of bee.

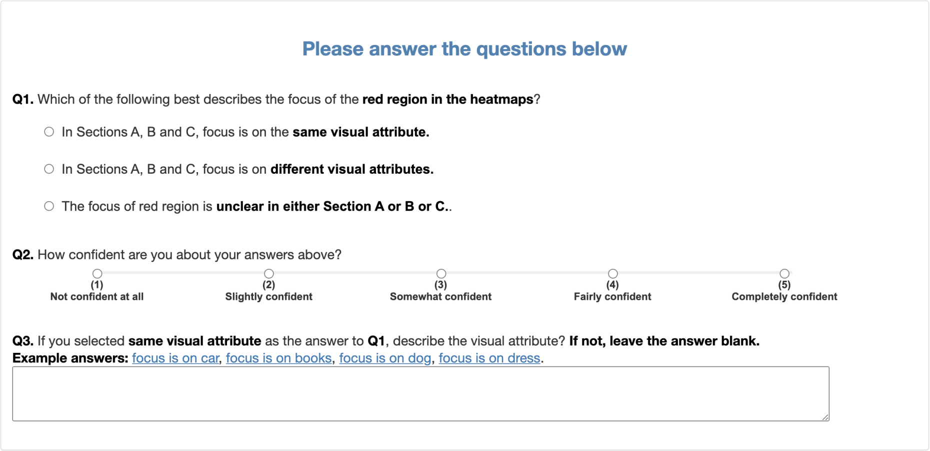
(c) Questionnaire

Figure 18: Mechanical Turk study for discovering spurious features. This figure is from Singla and Feizi [46] and included here for completeness.

that the accuracy drops experienced due to targeted corruption of input regions where certain features reside for a ResNet and a vision transformer are highly correlated too ($r^2 = 0.724$) (see figure 11). These results suggest that models are sensitive to the same data patterns and even rely on the different pieces of information (i.e. color, shape, texture) captured in the data patterns in the same ways.



(a) Heatmaps highlighting the visual attributes



Please answer the questions below

Q1. Which of the following best describes the focus of the **red region in the heatmaps**?

○ In Sections A, B and C, focus is on the **same visual attribute.**

○ In Sections A, B and C, focus is on **different visual attributes.**

○ The focus of red region is **unclear in either Section A or B or C.**.

Q2. How confident are you about your answers above?

○ (1) Not confident at all
○ (2) Slightly confident
○ (3) Somewhat confident
○ (4) Fairly confident
○ (5) Completely confident

Q3. If you selected **same visual attribute** as the answer to **Q1**, describe the visual attribute? **If not, leave the answer blank.**
Example answers: focus is on car, focus is on books, focus is on dog, focus is on dress.

(b) Questionnaire

Figure 19: Mechanical Turk study for validating heatmaps

To be perfectly clear, we do not claim to have captured *all* spurious (or core) dependencies, mainly because we only inspect the 5 most important robust neural features per class. We acknowledge that another reason for certain dependencies to be overlooked is because we use a single model to detect and measure the presence of concepts. However, this only occurs for patterns that other models are sensitive to but the adversarially trained model is not. The main patterns that fit this description are adversarial ones, which contain no semantic meaning and thus arguably *should* be overlooked for our purposes. In other work, adversarialy trained models were found to have increased reliance on spurious features [28]; this result was also corroborated in our spurious gap analysis. Thus, if anything, it seems like using an adversarially trained model may overlook core features, favoring spurious ones. Seeing as the goal of our analysis is to detect and assess spurious feature dependencies, we believe using robust neural features from an adversarially trained model is warranted, especially given the enhanced interpretability of this model, which allows for reliable heatmaps and feature attacks.

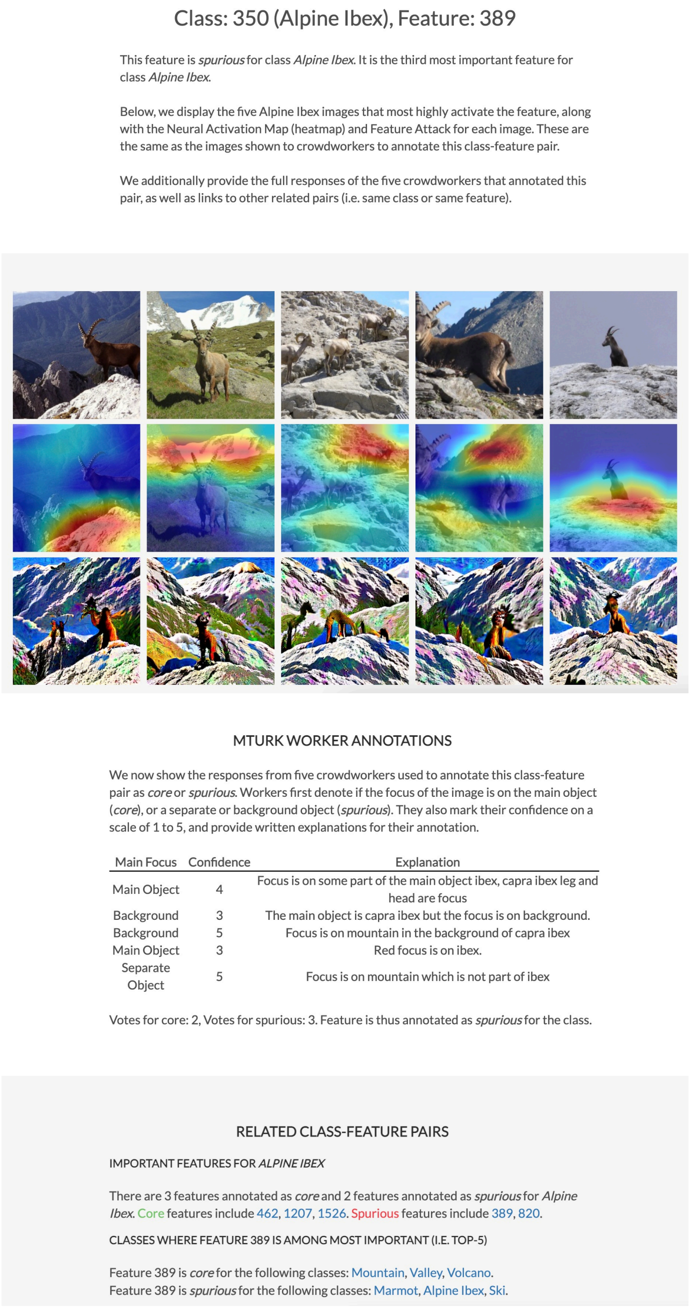
I.4 Our Contributions

We now detail the novel contributions we make to the Salient ImageNet framework and analysis from [46]. The most crucial contribution is the notion that to interpret and improve spurious correlation robustness, one can seek to simply reorganize the data they already have, as opposed to collecting more samples or additional instance-wise supervision (i.e. segmentations, manual annotations beyond class-label) that would require fundamental changes the training strategy. Using spuriosity rankings, we gain new and surprising insight into the nuanced ways that models rely on spurious features. Namely, we find that contrary to common belief, models can perform worse when spurious features are present than when they are absent (i.e. because of *spurious feature collision*, as some spurious features are depended on for multiple classes). Further, our spuriosity rankings naturally lends itself to a simple, effective, and efficient manner to both measure the biases caused by spurious feature reliance, and mitigate theses biases for any model. Finally, we make a crucial adjustment to the feature discovery framework of [46] that removes its most costly and at times prohibitive step (adversarial training). Observing that the framework can be applied by simply fitting a linear layer atop a frozen adversarially robust feature encoder vastly increases the usability of this framework, cutting the time and data required to perform feature discovery by huge margins.

We now take a closer look at our method for measuring bias caused by spurious feature reliance compared to the one proposed in [46], as we believe our method corrects crucial shortcomings of the prior work. Recall that the prior evaluation framework consisted of corrupting core or spurious regions by adding Gaussian noise multiplied by soft segmentations of core/spurious regions obtained via Neural Activation Maps of annotated robust neural features. The drop in accuracy due to these regional corruptions was used to measure the importance of the region. We enumerate the shortcomings of this prior framework, and explain how our method evades these concerns, below.

- **Introduction of a synthetic distribution shift.** The previous framework required taking images out of distribution by synthetically corrupting core and spurious regions with Gaussian noise. Model behavior on these non-natural images may not truly reflect how the model would behave on the more realistic distribution shift caused by breaking spurious correlations.
- **Instability.** Adding noise introduced hyperparameters, like the amplitude and norm of the noise, which we observe the metric to be undesirably sensitive to. In our method, the only hyperparameter is the number of images we include in the sets of highest and lowest spuriosity images. We verify our results our stable with respect to this hyperparameter.
- **Inability to compare across diverse models.** Because models have varying robustness to Gaussian noise (e.g. some more recent models, especially vision transformers which utilize heavier regularization to compensate for lacking inductive biases of convolutional networks, use noising as a data augmentation during training), drop in accuracy due to Gaussian noise cannot be reasonably compared as a means to measure how models use the corrupted image region. Our metric utilizes accuracy, which is standard and established for model comparisons. Moreover, we can appeal to the notion of effective robustness [27] to further control for varying accuracies of the models we consider.

Other contributions that are less novel but still impactful include massively expanding the scale of the feature annotation analysis and creating a web interface to view the discovered dependencies. Specifically, we annotate over $4\times$ more classes, yielding interpretation of feature dependencies for all 1000 classes in ImageNet. Given that this benchmark is ubiquitous, we believe a deep dissection of what one model perceives as important patterns in ImageNet can be extremely informative in better understanding how machine perception differs from human perception. We also perform the MTurk study to demonstrate the generalizability of our feature annotations for the top 20^{th} percentile of images based on feature activation as opposed to the top 5^{th} percentile, leading to reliable feature



Class: 350 (Alpine Ibex), Feature: 389

This feature is *spurious* for class *Alpine Ibex*. It is the third most important feature for class *Alpine Ibex*.

Below, we display the five Alpine Ibex images that most highly activate the feature, along with the Neural Activation Map (heatmap) and Feature Attack for each image. These are the same as the images shown to crowdworkers to annotate this class-feature pair.

We additionally provide the full responses of the five crowdworkers that annotated this pair, as well as links to other related pairs (i.e. same class or same feature).

MTURK WORKER ANNOTATIONS

We now show the responses from five crowdworkers used to annotate this class-feature pair as *core* or *spurious*. Workers first denote if the focus of the image is on the main object (*core*), or a separate or background object (*spurious*). They also mark their confidence on a scale of 1 to 5, and provide written explanations for their annotation.

Main Focus	Confidence	Explanation
Main Object	4	Focus is on some part of the main object ibex, capra ibex leg and head are focus
Background	3	The main object is capra ibex but the focus is on background.
Background	5	Focus is on mountain in the background of capra ibex
Main Object	3	Red focus is on ibex.
Separate Object	5	Focus is on mountain which is not part of ibex

Votes for core: 2, Votes for spurious: 3. Feature is thus annotated as *spurious* for the class.

RELATED CLASS-FEATURE PAIRS

IMPORTANT FEATURES FOR *ALPINE IBEX*

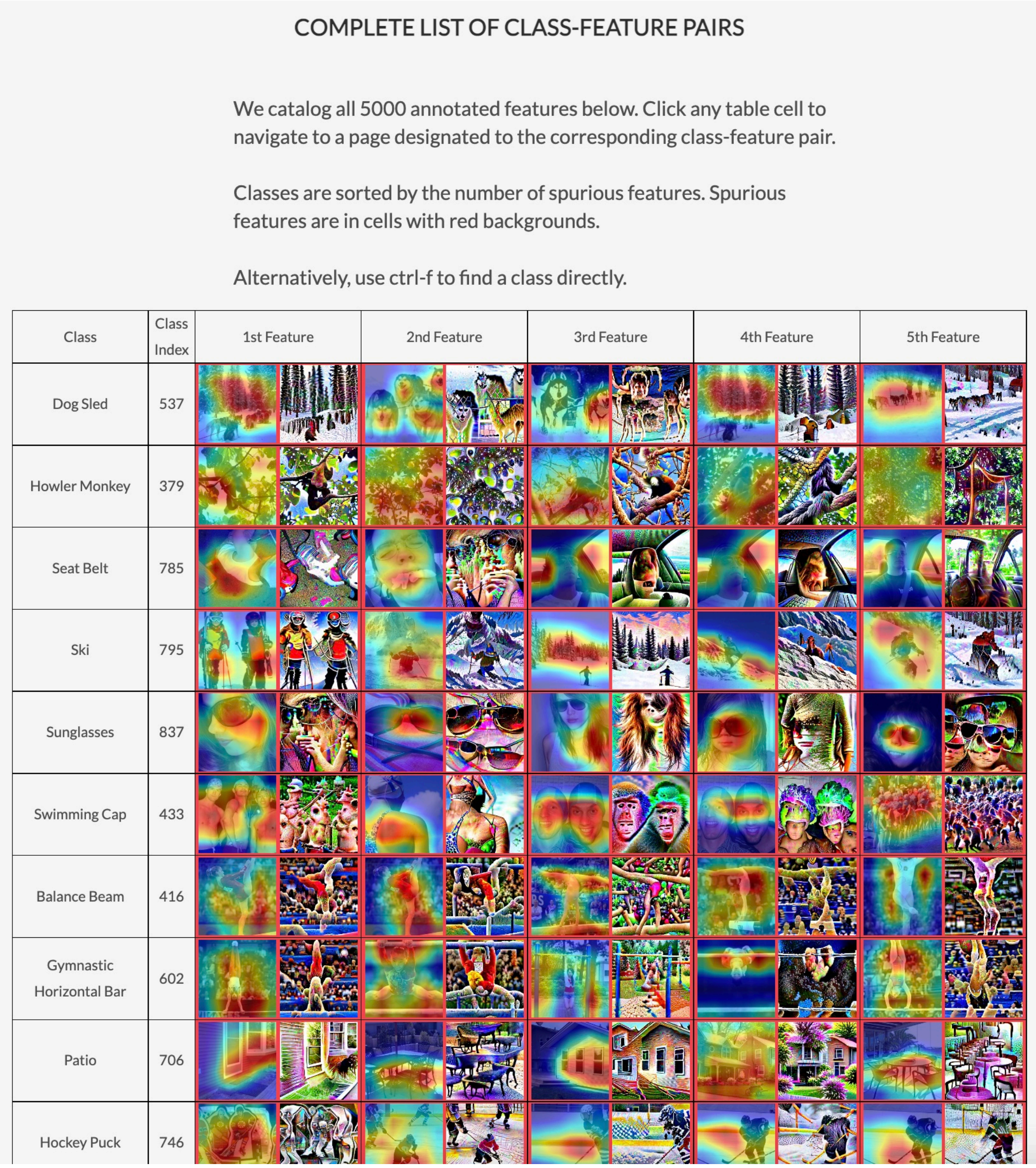
There are 3 features annotated as *core* and 2 features annotated as *spurious* for *Alpine Ibex*. Core features include 462, 1207, 1526. Spurious features include 389, 820.

CLASSES WHERE FEATURE 389 IS AMONG MOST IMPORTANT (I.E. TOP-5)

Feature 389 is *core* for the following classes: Mountain, Valley, Volcano.
Feature 389 is *spurious* for the following classes: Marmot, Alpine Ibex, Ski.

Figure 20: Example of a landing page for one class-feature dependency. We present all visualizations shown to MTurk workers to annotate the feature as cor or spurious, as well as the annotations and explanations given by the MTurk workers. Further, we provide links to pages for other features relevant to the class, and other classes that also rely on the feature.

soft segmentations for many more images. Lastly, we create a website, which, while simply a user-interface, will greatly increase ease of viewing and transparency into our analysis, as we show all the visualizations used to perform our core/spurious neural feature annotations. The website can be found at `https://salient-imagenet.cs.umd.edu`. We present screencaps of pages from the website in Figure 20 and Figure 21. As a web-interface, our work is now accessible to people who do not code or lack the resources to generate our visualizations on their own devices.



COMPLETE LIST OF CLASS-FEATURE PAIRS

We catalog all 5000 annotated features below. Click any table cell to navigate to a page designated to the corresponding class-feature pair.

Classes are sorted by the number of spurious features. Spurious features are in cells with red backgrounds.

Alternatively, use ctrl-f to find a class directly.

Class	Class Index	1st Feature	2nd Feature	3rd Feature	4th Feature	5th Feature
Dog Sled	537					
Howler Monkey	379					
Seat Belt	785					
Ski	795					
Sunglasses	837					
Swimming Cap	433					
Balance Beam	416					
Gymnastic Horizontal Bar	602					
Patio	706					
Hockey Puck	746					

Figure 21: Screencap of a main table from our web-interface for accessing all 5000 class-feature dependencies. One can simply search for a class by its name or ImageNet class index to see the five features deemed most important for (i.e. contributing the most to the logit of) the class. Each image then links to an individual landing page designated for the class-feature pair (see Figure 20 for an example).