Understanding Failures of Deep Networks via Robust Feature Extraction

Sahil Singla*
University of Maryland
ssingla@umd.edu

Besmira Nushi, Shital Shah, Ece Kamar, Eric Horvitz
Microsoft Research
{benushi,shitals,eckamar,horvitz}@microsoft.com

Abstract

Traditional evaluation metrics for learned models that report aggregate scores over a test set are insufficient for surfacing important and informative patterns of failure over features and instances. We introduce and study a method aimed at characterizing and explaining failures by identifying visual attributes whose presence or absence results in poor performance. In distinction to previous work that relies upon crowdsourced labels for visual attributes, we leverage the representation of a separate robust model to extract interpretable features and then harness these features to identify failure modes. We further propose a visualization method to enable humans to understand the semantic meaning encoded in such features and test the comprehensibility of the features. An evaluation of the methods on the ImageNet dataset demonstrates that: (i) the proposed workflow is effective for discovering important failure modes, (ii) the visualization techniques help humans to understand the extracted features, and (iii) the extracted insights can assist engineers with error analysis and debugging.

1. Introduction

It is critically important to understand the failure modes of machine learning systems, especially when they are employed in high-stakes applications in areas such as medicine, transportation, and security. Aggregate metrics in common use capture summary statistics on failure. While reporting overall performance is important, gaining an understanding of the specifics of failure is a core responsibility in the fielding of ML systems and components. For example, we need to understand situations where a self-driving car will fail to detect a pedestrian even when the system demonstrates high overall accuracy. Similarly, it is important to understand which features lead to misdiagnosis in chest x-rays even if the model has higher accuracy than humans. Such situation-specific understandings can guide the iterative process of model development and debugging.

Model performance can be wildly non-uniform for different clustering of instances and such heterogeneity is not reflected by standard metrics such as AUC or accuracy. For example, it was shown in [17] that a commercial model for emotion detection from facial expressions systematically failed for young children. Buolamwini et al. [3] found that gender detection in multiple commercial models had significantly higher error rates for women with darker skin tone. These examples highlight the importance of identifying natural clusters in the data with high failure rates. However, practical problems with these approaches still remain: (a) they require an expensive and time-consuming collection of metadata by humans, and (b) visual attributes that machine learning procedures pay attention to can be very different from the ones humans focus on (see Appendix Section F).

To resolve the aforementioned issues, we propose to leverage the internal representation of a robust model [23] to generate metadata. The key property that makes robust representations useful is that features can be visualized more easily than for a standard deep network [37, 12]. Our method called Barlow is inspired from Fault Tree Analysis [21] in safety engineering and uses robust representations as a building block. We shall illustrate the methods and demonstrate results on ImageNet dataset [10]. We find that the approach can be used to reveal two types of failures: Spurious correlations. A spurious correlation is a feature that is causally unrelated to the desired class but is likely to co-occur with the same class in the training/test data. For example, food is likely to co-occur with plates. However, the absence of the food from a plate image should not result in misclassification (see examples of spurious correlations in Figures 1a, 1b and 1e).

Overemphasized features. An overemphasized feature is a feature that is causally related to the desired class but the model gives excessive importance for classification (i.e., disregards other relevant features) and is unable to make a correct prediction when that feature is absent from the image. For example, a model may be likely to fail on a purse image if a buckle is absent or on a syringe when measure markings are absent (see Figures 1c and 1d).

Identifying these failure modes can provide valuable
guidance for improving model performance. For example, the identified failure modes in Figure 1 suggest the following interventions: (a) add images of maillot/monastery with diverse backgrounds, rhodesian ridgebacks without a collar (b) mask overemphasized features (buckle from purse, markings from syringe) in the training set.

In summary, we provide the following contributions:

1. An error analysis framework for discovering critical failure modes for a given model.
2. A feature extraction and visualization method based on robust model representations to enable humans to understand the semantic meaning of a learned feature.
3. A large-scale crowdsourcing study to evaluate the effectiveness of the visualization technique and the interpretability of robust feature representations.
4. A user study with engineers with experience using machine learning for vision tasks to evaluate the effectiveness of the methodology for model debugging.

2. Related work

Feature Visualization, Interpretability, and Robustness:
For a trained neural network, feature visualization creates images to either (i) maximize the neuron that we interested in or (ii) visualize the region that the neuron is paying attention to. Previous works [42, 41, 11, 24, 26, 44, 33, 32, 30, 35, 5] provide evidence that the internal representations of a neural network can capture important semantic concepts. However, as discussed in Olah et al. [29], for standard models, current feature visualization methods suffer from limitations. Recent work [37, 2, 23] shows that saliency maps are qualitatively more interpretable for robust models compared to standard models and align well with human perception. Moreover, Engstrom et al. [12] showed that, for robust models (in contrast to standard models), directly optimizing an image to maximize a certain neuron helps with visualizing the respective relevant learned features. Further theoretical investigation [2] for a two-layer ReLU network has shown that adversarial training purifies a small part of each learned feature after standard training. We note that disentangled representations [20, 16, 4, 18, 7, 22] based on VAEs [19] can also be used for feature extraction. However, it is difficult to scale these methods to large-scale, rich datasets such as ImageNet.

Failure explanation: In recent years, there has been increasing interest in the understandability of predictions made by machine-learned models [33, 34, 36, 1, 32]. Most of these efforts have focused on local explanations, where decisions about single images are inspected [12, 14, 32, 13, 6, 31, 38, 40]. In contrast, we focus on identifying major failure modes across the entire dataset and presenting them in a useful way to guide model improvement by actions such as fixing the data distribution and performing augmentation techniques. For this, we build upon previous work that discovers generalizable failure modes based on meta-data or features [27, 43, 8, 39]. These methods typically operate on tabular data where features are already available [8, 43], on language data with query-definable text operators [39], or on image data where features are collected from interme-
3. Background and method overview

Let $h : x \rightarrow y$ be a trained neural network that classifies an input image $x$ to one of the classes $y \in Y$. For a cluster of images $C$ in an overall benchmark $S$ (i.e., $C \subset S$), we use the following definitions to quantify failures:

**Definition 1** The error rate of a cluster is the portion of images in the cluster that are misclassified:

$$\text{ER}(C) = \frac{\sum_{x \in C} \mathbb{1}_{h(x) \neq y(x)}}{|C|}$$

**Definition 2** The error coverage of a cluster is the portion of all errors in the benchmark that fall in the cluster:

$$\text{EC}(C) = \frac{\sum_{x \in C} \mathbb{1}_{h(x) \neq y(x)}}{\sum_{x \in S} \mathbb{1}_{h(x) \neq y(x)}}$$

**Definition 3** The base error rate is the error rate for the whole benchmark, treating it as one cluster:

$$\text{BER} = \text{ER}(S)$$

We seek to describe failure cases in a benchmark $S$ by forming low-dimensional rules using a set of human-interpretable features which are a subset of all features $F$. For example, for an image recognition model that detects traffic lights, we want to be able to generate failure explanations such as “The error rate for detecting traffic lights is 20% higher when an image is captured in rainy weather and low light.”. The failure explanation in this case would be $\text{weather=rainy} \land \text{light_intensity=low}$. Such rules slice the data into clusters for which we can report metrics as in Definitions 1 and 2. Ideally, we would like to find clusters that jointly have large error rates and that cover a significant portion of the total errors from the benchmark. These criteria ensure that the explanatory rules will be of sufficient importance and generality. We note that the purpose of these explanations is not to predict failure but rather to provide actionable guidance to engineers via a small set of rules about failures and their indicators. The end-to-end Barlow workflow is depicted in Figure 2.

![Figure 2: Barlow error analysis workflow. Feature extraction and visualization are described in Section 4. Feature selection and failure mode generation in Section 5.](image)

### 4. Feature extraction and visualization

For each image in the set $S$, we first construct feature representation $F$ as a vector such that each element of the vector encodes some human-interpretable visual attribute. For this purpose, we use the penultimate layer (i.e., the layer adjacent to the logits layer) of a pretrained robust neural network to extract this feature vector. The robust model is trained via objective (1) for $l_2$ robustness.

$$\min_{\theta} \mathbb{E}_{(x,y) \sim D} \left[ \max_{\|x' - x\|_2 \leq \rho} \ell(h_\theta(x'), y) \right] \quad (1)$$

Tsipras et al. [37] show that for robust models, saliency maps are significantly more interpretable than standard models and align well with perceptually relevant features. Engstrom et al. [12] show that direct maximization of neurons of robust models is sufficient to generate easily recognizable visual attributes. Such output stands in contrast to the opacity of visualizations of features generated from standard models for which, even with regularization, visualized features are rarely human interpretable [29]. In Sections 8 and 9, we validate the earlier findings about human interpretability via real-world human studies.

Thus, for feature extraction, we use an adversarially trained Resnet-50 model [15] pretrained on ImageNet (using $\rho = 3$). We note that we can perform feature extraction using a robust model even when the model under inspection is not robust. In this case, we can consider the extracted features as attributes of the data and not necessarily as part of the representation employed within the model. Nevertheless, it is also important to note that these desirable properties of adversarially trained models make them more appealing for high-stakes applications where interpretability and debuggability are required.

We next describe our methodology for visualizing the sensitivity of a neuron in the representation layer of a robust model, consisting of three parts:

**Most activating images:** A common approach to visualizing a neuron’s sensitivity is to search through the given set of images to find top-$k$ instances that maximally activate the desired neuron. A challenge with this approach is...
that it does not identify the specific attributes of images that are responsible for the neuron activation versus correlations with the causes [29]. As an illustration, consider the example in first row of Figure 3. From these images, it is not clear whether the model is focused on sky, water, ground, or all of the above.

**Heatmaps:** To address the aforementioned problem, we propose to use heatmaps based on CAM [44] (second row in Figure 3), as an additional signal for visualization. For a Resnet-50 model, the representation layer is computed via a global average pooling operation to the tensor in the previous layer. We use the index of the desired neuron to retrieve the slice of the tensor from the previous layer at the same index (we refer to this as the feature map). Next, we normalize the feature map between 0 and 1 and resize it to match the input size. Details of this procedure are described in Appendix Section B. Figure 3 shows how heatmaps can help resolve the ambiguity.

**Feature Attack:** In some cases, heatmaps may not be sufficient to resolve ambiguity. For example, in Figure 4, it is not clear from the heatmaps whether the neuron is focused on the body, tail, or face of the monkey. Hence, we propose to maximize the neuron in the representation layer with respect to the original image (based on Engstrom et al. [12]). We observe that the limbs and tail (on top of green background) are highlighted and this resolves the ambiguity. We call the resulting visualization, a feature attack. More details are provided in the Appendix Section C.

We use the top-6 most activating images, corresponding heatmaps, and feature attack images to generate visualizations of the desired feature (Examples in Appendix Section G). We showcase the importance of these visualizations for interpretability via a user study detailed in Section 8.

### 5. Failure mode generation

We start with a set of features extracted from images using a robust model. Then, we explore the failure modes of a neural network (not necessarily the robust model) identified via consideration of the ground-truth labels. Our goal is to identify clusters with high error rates. In pursuit of failure analyses that can be understood with ease, we generate decision trees trained to predict failures (see Figure 2), determined by checking whether the model prediction is equal to the image label. Beyond binary classification of failure versus success, other labels of failure can be used, including characterizations derived from regression or localization procedures by specifying a threshold on how much deviation from the ground truth the application can tolerate.

We note that there are subtle challenges with relying on decision trees for explanations. Large numbers of features can make the process of finding a good split difficult due to the curse of dimensionality. From a usability perspective, failures for different classes can vary greatly and practitioners may be interested in understanding them separately [27]. Moreover, psychological studies provide evidence that people can only hold in memory and understand a limited number of chunks of information at the same time [25, 28, 9]. This suggests that human-in-the-loop methodologies such as Barlow need to avoid complex representations such as very large trees. Thus, we take the following steps:

**Class-based failure explanations.** We focus analyses and results on two types of class-based groupings: (i) prediction groupings and (ii) label groupings. For example, prediction groupings for the class “goldfish”, contain all images for which the model under inspection predicts “goldfish”. A failure analysis for this group enables us to understand false positives. Label groupings for the class “goldfish”, contain all images for which the ground truth in ImageNet is “goldfish” regardless of the model’s prediction. A failure analysis
5.1. Automatic evaluation of decision tree

Traditional metrics for evaluating the quality of a decision tree for a classification problem, such as accuracy, precision, and recall, are insufficient when the model is used for description and explanation. Ideally we would want to consider failure modes that include all possible failures in a benchmark. This goal is challenging because of (i) incompleteness in the feature set, (ii) difficulties in finding failure modes that generalize well for many examples at the same time, and, most importantly, (iii) certain failures may happen very rarely in the benchmark. Therefore, we focus on the explanatory properties of failure analyses.

For a given leaf node \( l \), we can use definitions 1 and 2 to define its leaf error rate \( \text{ER}(C_l) \), and leaf error coverage \( \text{EC}(C_l) \). \( C_l \) denotes the cluster of data that falls into leaf \( l \). For a decision tree \( T \), we then compute the following metric as the average leaf error rate (ALER).

**Definition 4** ALER of a tree \( T \) is the average error rate across all leaves weighed by the respective error coverage.

\[
\text{ALER}(T) = \sum_{l \in \text{leaves}(T)} \text{ER}(C_l) \times \text{EC}(C_l)
\]

Per the definition of the leaf error coverage, we have \( \sum_{l \in \text{leaves}(T)} \text{ER}(C_l) = 1 \) because all failure instances will be covered by exactly one leaf.

As an example, consider the simple 1-depth decision tree given in Figure 5. Since the leaf error rate is less than 0.5 for both leaf nodes, tree precision and tree recall are both zero. However, since the error rate of leaf 1 is significantly higher than the base error rate (0.454 >> 0.2), leaf 1 is important as a failure mode description (and more predictive than leaf 2), which is exactly the notion we seek to capture for describing high concentrations of error.

The key property that makes ALER useful is that if it is equal to some quantity \( q \), then there is at least one leaf in the decision tree with leaf error rate greater than \( q \).

**Proposition 1** For a tree \( T \) with \( \text{ALER}(T) = q \), there exists at least one leaf \( l \), with leaf error rate \( \text{ER}(C_l) \geq q \).

The proposition follows from the fact that all weights (i.e., leaf error coverage) are less than or equal to 1. For the decision tree in Figure 5, \( \text{BER} = 0.291 \), which is greater than the base error rate 0.2 by a margin of 0.091. This signals the presence of a leaf with error rate of at least 0.291 which we know is leaf 1, \( \text{ER}(C_1) = 0.454 \).

Since the root node of the tree already comes with a prior on the error rate (BER), we are interested in how much more value the discovered failure modes in the tree add when compared to the root. Thus, for the automated evaluation we use ALER − BER to measure the increase in error and discrepancy that the tree explains.

This metric also suggests that leaves with higher value of \( \text{ER}(C_l) \times \text{EC}(C_l) \) contribute more to ALER and are thus more important for explaining failures. This leads to the following metric for ranking nodes for failure explanation:

**Definition 5** The Importance Value i.e \( \text{IV}(C_l) \) of a leaf node \( l \) in a tree \( T \) is defined as:

\[
\text{IV}(C_l) = \text{ER}(C_l) \times \text{EC}(C_l)
\]

6. Failure modes discovered by Barlow

We now describe some failure modes discovered by Barlow when analyzing a standard Resnet-50 model, using a robust Resnet-50 model for feature extraction (both trained on ImageNet). We use the ImageNet training set (instead of the validation set) for failure analysis due to the larger

![Figure 5: Illustration of ALER metric. For both leaf nodes, precision and recall are 0 since \( \text{ER} < 0.5 \) however leaf 1 is more important for failure mode discovery.](image)
number of image instances per class (1300 vs. 50 in the validation set). Note that from a methodology perspective, practitioners may apply the Barlow workflow to any benchmark with ample data and failures, but the failure modes that they discover may be different depending on the nature of the benchmark. For ease of exposition, all decision trees have depth one. We selected the leaf node with highest $IV(C_l)$ and visualized the feature used for arriving at this node. More examples of failure modes are in Appendix Section F.1 (using a standard model) and Section F.2 (using a robust model). We will provide public access to data on results of the error analysis for all 1000 ImageNet classes for label and prediction grouping.

**Label grouping** In Figure 6, the top row shows the most highly activating images for a feature identified important for failure. The heatmap and feature attack provide strong evidence that the model is paying attention to the buckle on purses. The bottom row shows randomly selected failures in this leaf node. We do not observe a buckle in any of these images even though they have ground-truth labels of purse indicating the feature (while correlated, not causal) is important for making correct predictions. In other words, it appears that, whenever the purse image does include a buckle, it is more difficult for the model to predict it correctly as a purse (error increases by 10.94%) and the prediction is more likely to be a false negative.

**Prediction grouping** Figure 7 displays an example where the heatmap and feature attack provide strong evidence that the model is paying attention to the collar of the dog. In the bottom row (i.e randomly selected failure images), we do not observe a dog collar in any of these images even though the model predicts a rhodesian ridgeback for all of them. This indicates that the feature (correlated not causal) is important for making correct predictions. In other words, whenever the model predicts rhodesian ridgeback, but the image does not contain a dog collar, the prediction is more likely to be a false positive (error increases by 10.91%).
7. Experiments with automated evaluation

We now report on a study of factors that influence the effectiveness of error analysis: decision tree depth, robustness of model, grouping strategy. We train decision trees with depths of 1 and 3 for each model and grouping strategy. For evaluating a decision tree, we use the metric $\text{ALER} - \text{BER}$ as defined in Section 5.1. We also select the leaf with highest importance value $\text{IV}(C_l)$ for each decision tree and evaluate whether the cluster of data in this leaf satisfies the two conditions: $\text{ER}(C_l) > \text{BER} + \rho$ and $\text{EC}(C_l) > \tau$, with $\rho = 0.1$ and $\tau = 0.2$. In Appendix Table 1, we report for each model, grouping strategy, and tree depth the fraction of such valid leaves across all 1000 classes that satisfy these conditions.

In summary, we make the following observations:

• Grouping by ground-truth labels results in better decision trees (by $\text{ALER} - \text{BER}$ score) compared to prediction grouping for both standard and robust models (Figure 8), and also for decision trees with different depths.

• Failure explanation for a robust model results in significantly better score compared to standard model for both grouping strategies and depths of decision tree (See Appendix Figures 17, 20).

• Increasing depth significantly improves the score for both models as shown in Appendix Figures 18 and 21.

• For the standard model, on all trees (except prediction grouping and depth=1 where it is 0.211) the fraction of classes with at least one valid leaf is at least 0.596. For the robust model, on all trees the fraction of classes with at least one valid leaf is at least 0.787.

8. Crowd study on feature interpretability

To understand the effectiveness of the feature visualization method, we conducted a crowdworker study using Amazon Mechanical Turk (MTurk). For both grouping strategies (prediction and label), we selected the top, middle, and bottom 20 classes (total 60 x 2) based on the error rate of the robust Resnet-50 and selected the top 10 features with the highest mutual information on failure, resulting in 1200 visualizations and 5971 human answers. In total, we had 312 unique workers, each completing on average 19.14 tasks. 20% of the workers completed at least 10 tasks. All visualizations were evaluated by five workers, except 29 for which only had four assessments. Workers were paid $0.5 per hit, with an average salary of $12 per hour. Anonymized data from this study will be made available.

For each visualization, workers were shown three sections: A (most activating images), B (heatmaps), and C (feature attack), and asked to answer the set of questions in Figure 66 in the Appendix Section G. The questions were designed with two goals: (1) to collect human-generated feature descriptions, and (2) to evaluate the ease with which workers can understand and describe the features.

Figure 9a shows the cumulative distribution of answers to Questions 3 (Ease of understanding) and 4 (Confidence). We observe that workers are able to describe visualizations with confidence and ease (average likert scale > 3) for a large number of features (> 800). Figure 9b, shows workers’ preferences between heatmaps and feature attacks (Question 5). In response to a question about the most useful views, 43.18% answers (out of 5971) report Section B (heatmap) as most useful, 17.67% report Section C (feature attack), and 32.14% report both Section B and C. Only 7% of the answers reported None. These results provide evidence that heatmaps are most valuable in contributing to the understanding of the meaning of features, and that feature attack visualizations results are also valuable individually and together with heatmaps. Importantly, only 0.92% of all 1200 features received None as the majority vote from all 5 workers. This provides further evidence that our methodology is effective in explaining a large number of features.

Figure 10 shows the cumulative distribution of worker agreement on the textual feature descriptions in crowd study.
the score can capture common themes in descriptions even when workers use different words but with similar meaning (e.g., digit vs. number). We observe that agreement increases with longer descriptions. Qualitatively, we see that agreement is higher ($\geq 0.45$) when the images in the visualization contain fewer objects and the objects are salient. Sample descriptions from workers along with agreement scores can be found in Appendix Section G. We further emphasize that since these workers do not routinely visualize and debug models using such visualizations, our results are likely to be a lower bound on the true effectiveness of our visualization methodology for engineers.

9. Study with machine learning practitioners

To evaluate the usefulness of Barlow for error analysis and debugging, we conducted user studies with 14 ML practitioners at Microsoft. Participants were recruited via four mailing lists on the topics of “Machine Learning” and “Computer Vision”. Participation requirements included previous experience in applying machine learning on vision tasks. A summary of roles and years of experience is shown in Table 8 in the Appendix Section H. We aimed for recruiting practitioners that are frequently involved in model building and evaluation, who would be the primary audience for Barlow.

Study Protocol: Each study lasted one hour and started with a description of the Barlow workflow and terminology as shown in Figure 2. We asked participants to imagine being positioned in the following scenario:

“We will conduct error analysis for a classification model (robust Resnet-50) trained on ImageNet. We ask you to imagine that you are part of a team that will deploy this in production and wants to understand where the model is incorrect and identify action items upon failures.”

Each participant inspected two different class groupings. The first grouping was randomly assigned from a set of five pre-selected groupings where the most important features for failure explanation were considered as “easy to describe” by MTurk workers to facilitate onboarding (exact assignment shown in Table 9, Appendix Section H). The second grouping was selected randomly. For each grouping, participants were first presented with a decision tree (of depth 2) describing the failure modes. For each node, they could see the respective error rate and coverage, the instances in the node, and the visualization of the feature responsible for the split. To collect feedback on their experience, we asked the following questions:

Q1: Can you describe the cluster of images defined by this feature? All participants were able to describe the features of at least one of the two groupings they inspected. 10 out of 14 participants were able to describe features of both groupings. The four participants who could not describe the features of one grouping faced one or both of the following issues: either the feature represented two or more concepts (e.g., sky and wires), or they had a difficult time to properly understand the sensitivity of the feature’s visual appearance to its activation value. In such cases, they preferred to see more than five examples for refining their hypothesis.

Q2: Is the feature necessary for the task or do you think it is a spurious correlation? Participants were able to identify several surprising spurious correlations (e.g., “This feature looks like a hair detector, should not be used for Seat Belts.” - P12, “Seems like the model can do well only when the photos of Water Jugs are taken either professionally or on a clean background.” - P17). Similarly, they could identify when the failure modes were related to necessary but not sufficient features (e.g., “The model is focusing on the face of the dog. We might also need to make the model look at the legs, hair length, hair color, etc.” - P7).

Q3: What action items would you take for mitigating the errors for this class? In overall, participants reported that the identified failure explanations gave them ideas about how to continue with mitigating the errors, given sufficient time and resources. Data Collection was the most popular action item (Figure 11) either for mitigating lack of data diversity or addressing statistical biases (e.g. “I would check if most Tiger Cats in the training data have green background and if so, add more diverse photos.” - P5). Changes in model architecture were most often related to increasing the capacity of the model (e.g., “The model is using the same
features for detecting zebra patterns and spatulas, perhaps it’s best to use more than one feature for this.” - P5). Label correction was suggested when participants thought that the model’s prediction was not completely wrong or when certain true positive examples in the training data were confusing (e.g. “Some hooks are not really hooks.” - P12).

**User satisfaction scores:** The study ended with a survey of six agreement statements, which the participants completed not in the presence of the interviewer (results in Figure 12). We observe overall enthusiasm about the usefulness of the workflow and its ability to surface important failures (e.g. “The approach can help make DNNs much more explainable, making such analysis mandatory before deploying a solution. Perhaps, a step further could be to take a similar test with humans to ensure the model has learned explainable features, before we ship the model.” - P7).

## 10. Conclusion and future work

We proposed methods and a workflow aimed at providing insights about critical failures of machine learning models by using features extracted from robust representations. Beyond developing and refining the methodology, we performed two sets of studies with human subjects, focusing on the interpretability of features and on the usefulness of the error analysis in realistic settings. We showed that the practitioners can effectively leverage the methods to interpret the features and identify most significant clusters in data with errors due to issues such as systematic spurious correlations. Such analyses hold the promise for supporting cycles of iterative improvement that can mitigate failures without requiring expensive manual metadata collection. We envision promising directions for future research on refining methods for hypothesis testing via the consideration of counterfactuals based on making specific, understandable changes to the data and models. We see opportunities ahead more generally at the intersection of methods for robust machine learning and generative models that may help synthesize independent and interpretable features to facilitate model debugging.

## 11. Acknowledgments

We would like to thank Parham Mohadjer for helping with the user study integration, Nathan Cross for helpful discussions on extending Barlow to radiology images, Soheil Feizi for giving useful feedback on the paper and throughout the project, as well as all participants in our user studies whose feedback significantly refined this work. Finally, we also want to acknowledge the contributions of Arda Deniz Eden, the newest team member of our group.

## References

[1] Julius Adebayo, J. Gilmer, M. Muelly, Ian J. Goodfellow, M. Hardt, and Been Kim. Sanity checks for saliency maps. In NeurIPS, 2018. 2
[2] Zeyuan Allen-Zhu and Yuanzhi Li. Feature purification: How adversarial training performs robust deep learning. arXiv preprint arXiv:2005.10190, 2020. 2
[3] Joy Buolamwini and Timnit Gebru. Gender shades: Intersectional accuracy disparities in commercial gender classification. In FAT, 2018. 1
[4] Christopher P. Burges, Irina Higgins, Arka Pal, Loic Matthey, Nick Watters, Guillaume Desjardins, and Alexander Lerchner. Understanding disentangling in $\beta$-vae. In NeurIPS, 2017. 2
[5] Shan Carter, Zan Armstrong, Ludwig Schubert, Ian Johnson, and Chris Olah. Activation atlas. Distill, 2019. https://distill.pub/2019/activation-atlas. 2
[6] Chun-Hao Chang, Elliot Creager, Anna Goldenberg, and David Duhemaud. Explaining image classifiers by counterfactual generation. In International Conference on Learning Representations, 2019. 2
[7] Ricky T. Q. Chen, Xuechen Li, Roger Grosse, and David Duhemaud. Isolating sources of disentanglement in variational autoencoders. In NeurIPS, 2018. 2
[8] Yeunoh Chung, Tim Kraska, Neoklis Polyzotis, Ki Hyun Tae, and Steven Euijiong Whang. Slice finder: Automated data slicing for model validation. In 2019 IEEE 35th International Conference on Data Engineering (ICDE), pages 1550–1553. IEEE, 2019. 2
[9] Nelson Cowan. The magical number 4 in short-term memory: A reconsideration of mental storage capacity. Behavioral and brain sciences, 24(1):87–114, 2001. 4
[10] J. Deng, Wei Dong, R. Socher, L. Li, K. Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. 2009 IEEE Conference on Computer Vision and Pattern Recognition, pages 248–255, 2009. 1
[11] Alexey Dosovitskiy and Thomas Brox. Inverting visual representations with convolutional networks. In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2016. 2
[12] Logan Engstrom, Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Brandon Tran, and Aleksander Madry. Adversarial robustness as a prior for learned representations, 2019. 1, 2, 3, 4
[13] Yash Goyal, Amir Farid, U. Shalit, and Been Kim. Explaining classifiers with causal concept effect (cace). ArXiv, abs/1907.0165, 2019. 2
[14] Yash Goyal, Ziyian Wu, Jan Ernst, Dhruv Batra, Devi Parikh, and Stefan Lee. Counterfactual visual explanations. In ICML, 2019. 2
[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 770–778, 2015. 3
[16] I. Higgins, Loic Matthey, A. Pal, C. Burges, Xavier Glorot, M. Botvinick, S. Mohamed, and Alexander Lerchner. beta
vae: Learning basic visual concepts with a constrained variational framework. In ICLR, 2017.

[17] Ayanna Howard, Cha Zhang, and Eric Horvitz. Addressing bias in machine learning algorithms: A pilot study on emotion recognition for intelligent systems. In IEEE Workshop on Advanced Robotics and its Social Impacts, 2017.

[18] Hyunjik Kim and Andriy Mnih. Disentangling by factorising. In Jennifer Dy and Andrea Krause, editors, Proceedings of the 35th International Conference on Machine Learning, volume 80 of Proceedings of Machine Learning Research, pages 2649–2658, Stockholmsmässan, Stockholm Sweden, 10–15 Jul 2018. PMLR.

[19] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In ICLR, 2014.

[20] Tejas D. Kulkarni, Will Whitney, Pushmeet Kohli, and Joshua B. Tenenbaum. Deep convolutional inverse graphics network. In NIPS, 2015.

[21] W. S. Lee, D. L. Grosh, F. A. Tillman, and C. H. Lie. Fault tree analysis, methods, and applications: A review. IEEE Transactions on Reliability, R-34(3):194–203, Aug 1985.

[22] Francesco Locatello, Stefan Bauer, Mario Lucic, Gunnar Raetsch, Sylvain Gelly, Bernhard Schölkopf, and Olivier Bachem. Challenging common assumptions in the unsupervised learning of disentangled representations. In ICML, 2019.

[23] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In ICLR, 2018.

[24] Aravindh Mahendran and A. Vedaldi. Visualizing deep convolutional neural networks using natural pre-images. International Journal of Computer Vision, 120:233–255, 2016.

[25] George A Miller. The magical number seven, plus or minus two: Some limits on our capacity for processing information. Psychological review, 63(2):81, 1956.

[26] Anh Mai Nguyen, Jason Yosinski, and Jeff Clune. Multi-faceted feature visualization: Uncovering the different types of features learned by each neuron in deep neural networks. In ICML Workshop on Visualization for Deep Learning, 2016.

[27] Besmira Nushi, Ece Kamar, and Eric Horvitz. Towards accountable AI: hybrid human-machine analyses for characterizing system failure. In Yiling Chen and Gabriella Kazai, editors, Proceedings of the Sixth AAAI Conference on Human Computation and Crowdsourcing, HCOMP, pages 126–135. AAAI Press, 2018.

[28] Klaus Oberauer and Laura Hein. Attention to information in working memory. Current Directions in Psychological Science, 21(3):164–169, 2012.

[29] Chris Olah, Alexander Mordvintsev, and Ludwig Schubert. Feature visualization. Distill, 2017. https://distill.pub/2017/feature-visualization.

[30] Chris Olah, Arvind Satyanarayan, Ian Johnson, Shan Carter, Ludwig Schubert, Katherine Ye, and Alexander Mordvintsev. The building blocks of interpretability. Distill, 2018. https://distill.pub/2018/building-blocks.

[31] Matthew O’Shaughnessy, Gregory Canal, Marissa Connor, Mark Davenport, and Christopher Rozell. Generative causal explanations of black-box classifiers. In NeurIPS, 2019.

[32] R. R. Selvaraju, Abhishek Das, Ramakrishna Vedantam, Michael Cogswell, D. Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. International Journal of Computer Vision, 128:336–359, 2019.

[33] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. In Workshop at International Conference on Learning Representations, 2014.

[34] Daniel Smilkov, Nikhil Thorat, Been Kim, Fernando B. Viégas, and Martin Wattenberg. Smoothgrad: removing noise by adding noise. In ICML Workshop on Visualization for Deep Learning, 2017.

[35] Pascal Sturmfels, Scott Lundberg, and Su-In Lee. Visualizing the impact of feature attribution baselines. Distill, 2020. https://distill.pub/2020/attrition-baselines.

[36] Mukund Sundararajan, Ankur Taly, and Qi Qi Yan. Axiomatic attribution for deep networks. In ICML, 2017.

[37] Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. Robustness may be at odds with accuracy. In ICLR, 2018.

[38] Sahil Verma, John Dickerson, and Keegan Hines. Counterfactual explanations for machine learning: A review, 2020.

[39] Tongshuang Wu, Marco Tulio Ribeiro, Jeffrey Heer, and Daniel S. Weld. Errudite: Scalable, reproducible, and testable error analysis. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 747–763, 2019.

[40] Chih-Kuan Yeh, Cheng-Yu Hsieh, Arun Sai Suggala, David I. Inouye, and Pradeep D. Ravikumar. On the (in)fidelity and sensitivity of explanations. In NeurIPS, 2019.

[41] Jason Yosinski, Jeff Clune, Anh Mai Nguyen, Thomas J. Fuchs, and Hod Lipson. Understanding neural networks through deep visualization. In ICML Deep Learning Workshop, 2016.

[42] Matthew D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. In ECCV, 2014.

[43] Jiawei Zhang, Yang Wang, Piero Molino, Lezhi Li, and David S Ebert. Manifold: A model-agnostic framework for interpretation and diagnosis of machine learning models. IEEE transactions on visualization and computer graphics, 25(1):364–373, 2018.

[44] B. Zhou, A. Khosla, Lapedriza. A., A. Oliva, and A. Torralba. Learning Deep Features for Discriminative Localization. In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.
A. Feature extraction

Figure 13 shows our feature extraction mechanism. We use an adversarially trained Resnet-50 model (threat model is an $l_2$ ball of radius 3). For feature extraction, we use the penultimate layer i.e layer adjacent to the logits layer (also the output of global average pooling layer for a Resnet-50 architecture). In practice, in order to extract these features for a given benchmark, we run each image in the benchmark through the model (in inference time) and use the activation in this layer as feature values.

B. Heatmap generation

Figure 14 describes the heatmap generation method. We select the feature map from the output of the tensor of the previous layer (i.e before applying the global average pooling operation). Next, we normalize the feature map between 0 and 1 and resize the feature map to match the image size.

C. Feature attack

In Figure 15, we illustrate the process behind the feature attack. We select the feature we are interested in and optimize the image to maximize its value to generate the visualization. $\epsilon$ is a hyperparameter used to control the amount of change allowed in the image. For optimization, we use gradient ascent with step size = 1, $\epsilon = 500$ and number of iterations = 500.

D. Failure mode generation

We describe our procedure for generating failure modes in Algorithm 1. The algorithm can take as an input any cluster of image data $C$. In our experiments, the clusters were defined via image grouping by label and model prediction. However, practitioners may choose to apply the same procedure to clusters of images defined in other ways such as for example pairs of classes that are often confused with each other or unions of prediction and label groupings for the same class.

Algorithm 1: Failure mode generation procedure.

\begin{itemize}
  \item Input: features: $F$, model: $h$, image cluster: $C$, number of features: $k$, tree parameters: $A$, error rate threshold: $\rho$, error coverage threshold: $\tau$
  \item Output: leaves with high error concentration: $L$
  \end{itemize}

$L = \emptyset$

BER = ER($C$)

$E(x) = \begin{cases} 0 & h(x) = y \\ 1 & h(x) \neq y \end{cases}$ \forall(x, y) \in C$

$F^* = \emptyset$

while $|F^*| < k$ do

$F^* = F^* \cup \arg\max_{f \in F \setminus F^*} IG(E; f)$

end

$T = \text{train\_decision\_tree}(F^*, E, A)$ for $l \in T$ do

if (ER($C_l$) > BER + $\rho$) and (EC($C_l$) > $\tau$) then

$L = L \cup \{l\}$

end

end

Return: $L$

E. Automatic evaluation of decision tree

We now report on a study of factors that influence the effectiveness of error analysis: decision tree depth, robustness of model, grouping strategy. We train decision trees with depths of 1 and 3 for each model and grouping strategy. For
Table 1: For each model, grouping strategy and decision tree depth we report the fraction of *valid leaves* across all 1000 classes, i.e the leaf nodes that satisfy these two conditions:
\[ \text{ER}(C_l) > \text{BER} + \rho \text{ and } \text{EC}(C_l) > \tau, \]
with \( \rho = 0.1 \) and \( \tau = 0.2 \) in the last column. Semantically, these would be all leaves with an error increase of at least 10% that cover 20% of the failures or more.

| Model   | Depth | Grouping | Fraction |
|---------|-------|----------|----------|
| Standard| 1     | Label    | 0.596    |
| Standard| 1     | Prediction| 0.211   |
| Standard| 3     | Label    | 0.900    |
| Standard| 3     | Prediction| 0.649   |
| Robust  | 1     | Label    | 0.977    |
| Robust  | 1     | Prediction| 0.787   |
| Robust  | 3     | Label    | 0.899    |
| Robust  | 3     | Prediction| 0.804   |

evaluating a decision tree, we use the metric \( \text{ALER} - \text{BER} \) as defined in Maintext Section 5.1, Definition 4. We also select the leaf with highest importance value \( \text{IV}(C_l) \) for each decision tree (Maintext Definition 5) and evaluate whether the cluster of data in this leaf satisfies the two conditions:
\[ \text{ER}(C_l) > \text{BER} + \rho \text{ and } \text{EC}(C_l) > \tau, \]
with \( \rho = 0.1 \) and \( \tau = 0.2 \). In Table 1, we report for each model, grouping strategy, and tree depth the fraction of such *valid leaves* across all 1000 classes that satisfy these conditions.

We make the following observations:

- Grouping by ground-truth labels results in better decision trees (by \( \text{ALER} - \text{BER} \) score) compared to prediction grouping for both standard and robust models and also for decision trees with different depths. This is true even when \( \text{BER} \) is similar (See Figures 16 and 19).
- Failure explanation for a robust model results in significantly better score compared to standard model for both grouping strategies and depths of decision tree. This is again true, even when \( \text{BER} \) is similar (See Figures 17 and 20). While this observation is intuitive, given that the extracted features come from the robust model, it serves as an additional motivation for employing robust models in practice. The evaluation shows that such models might simplify the debugging and error analysis processes.
Figure 16: Comparison between grouping strategies using a decision tree of depth 1.

Figure 17: Comparison between standard and robust models using a decision tree of depth 1.

Figure 18: Comparison between decision trees of different depths using a standard model.

Figure 19: Comparison between grouping strategies using a decision tree of depth 3.

Figure 20: Comparison between standard and robust models using a decision tree of depth 3.

Figure 21: Comparison between decision trees of different depths using a robust model.
In this section, we describe several failure modes discovered by Barlow. For experiments in subsection F.1, we analyze the errors of a standard Resnet-50 model for failure analysis and for subsection F.2, we inspect a robust Resnet-50 model. In both cases, we use a robust Resnet-50 model for feature extraction. All models were pretrained on ImageNet. We use the ImageNet training set (instead of the validation set) for failure analysis due to the larger number of instances and failures. For ease of exposition, all decision trees have depth one. We select the leaf node with highest Importance Value (i.e IV as defined in Definition 5) for visualizing the failure mode. Since the tree has depth one, we can visualize the one feature that defines this leaf node.

All feature visualizations are organized as follows. The topmost row shows the most activating images. The second row shows the heatmaps. The third row shows feature attack images. Finally, the bottom row shows randomly selected failure examples in the leaf node.

For all tables, BER denotes the Base Error Rate, ER denotes Error Rate, EC denotes Error Coverage for the leaf with highest Importance Value and ALER denotes Average Leaf Error Rate.

F.1. Failure explanation for a standard model

F.1.1 Grouping by label

Results are in Table 2.

F.1.2 Grouping by prediction

Results are in Table 3.

F.2. Failure explanation for a robust model

F.2.1 Grouping by label

Results are in Table 4.

F.2.2 Grouping by prediction

Results are in Table 5.
Table 2: Results on a standard Resnet-50 model using grouping by label.
Figure 23: Visualization of feature[995]. For images with label monastery, when feature[995] < 0.1428, error rate increases to 0.6345 (+24.84%).

Figure 24: Visualization of feature[1365]. For images with label maillot, when feature[1365] > 0.7066, error rate increases to 0.7564 (+9.72%).
Figure 25: Visualization of feature[1679]. For images with label monitor, when feature[1679] < 0.8030, error rate increases to 0.6061 (±13.00%).

Figure 26: Visualization of feature[544]. For images with label tiger cat, when feature[544] < 0.2036, error rate increases to 0.8754 (±37.85%).
Figure 27: Visualization of feature[1911]. For images with label titi, when feature[1911] < 0.7329, error rate increases to 0.5240 (+11.09%).

Figure 28: Visualization of feature[776]. For images with label lotion, when feature[776] < 0.3313, error rate increases to 0.4797 (+11.73%).
Figure 29: Visualization of feature[1378]. For images with label pitcher, when feature[1378] < 0.7671, error rate increases to 0.6253 (+28.15%).

Figure 30: Visualization of feature[1611]. For images with label hog, when feature[1611] < 0.0578, error rate increases to 0.6842 (+35.27%).
Figure 31: Visualization of feature[1264]. For images with label trench coat, when feature[1264] < 0.6915, error rate increases to 0.3196 (+18.57%).

Figure 32: Visualization of feature[1081]. For images with label baseball, when feature[1081] < 0.5461, error rate increases to 0.3034 (+19.65%).
Figure 33: Visualization of feature 443. For images with prediction bakery, when feature 443 < 1.1382, error rate increases to 0.3875 (+11.80%).

| Class name          | Feature index | Decision rule | BER   | ER   | EC   | ALER | Feature visualization | Feature name (from visualization) |
|---------------------|---------------|---------------|-------|------|------|------|-----------------------|-----------------------------------|
| bakery              | 443           | < 1.1382      | 0.2695| 0.3875| 0.7793| 0.3307| Figure 33             | shelves with sweets                |
| polaroid camera     | 793           | < 0.8166      | 0.1141| 0.2713| 0.8671| 0.2384| Figure 34             | close-up view of camera            |
| saluki              | 1395          | < 0.3263      | 0.1122| 0.2287| 0.5772| 0.1600| Figure 35             | long and hairy dog ears            |
| trailer truck       | 1451          | < 0.2181      | 0.1121| 0.2184| 0.5036| 0.1472| Figure 36             | white truck                        |
| apiary              | 1909          | < 0.5646      | 0.1057| 0.2341| 0.8041| 0.1969| Figure 37             | white boxes                        |
| anemone fish        | 262           | < 0.1792      | 0.1056| 0.2573| 0.4247| 0.1516| Figure 38             | red fish                           |
| theater curtain     | 1063          | < 0.9047      | 0.1049| 0.2482| 0.5072| 0.1583| Figure 39             | red curtain                        |
| forklift            | 943           | < 1.1721      | 0.1047| 0.2379| 0.8889| 0.2136| Figure 40             | orange car                         |
| french bulldog      | 404           | < 0.2946      | 0.1022| 0.2103| 0.3712| 0.1273| Figure 41             | dog nose                           |
| syringe             | 638           | < 0.2325      | 0.2020| 0.3519| 0.4894| 0.2455| Figure 42             | measurements                       |
| rhodesian ridgeback | 1634          | < 1.3779      | 0.2093| 0.3184| 0.4561| 0.2337| Figure 43             | dog collar                         |

Table 3: Results on a standard Resnet-50 model using grouping by prediction.
Figure 34: Visualization of feature[793]. For images with prediction polaroid camera, when feature[793] < 0.8166, error rate increases to 0.2713 (15.72\%).

Figure 35: Visualization of feature[1395]. For images with prediction saluki, when feature[1395] < 0.3263, error rate increases to 0.2287 (11.65\%).
Figure 36: Visualization of feature[1451]. For images with prediction trailer truck, when feature[1451] < 0.2181, error rate increases to 0.2184 (+10.63%).

Figure 37: Visualization of feature[1909]. For images with prediction apiary, when feature[1909] < 0.5646, error rate increases to 0.2371 (+13.14%).
Figure 38: Visualization of feature[262]. For images with **prediction anemone fish**, when feature[262] < 0.1792, error rate increases to 0.2573 (**+15.17%**).

Figure 39: Visualization of feature[1063]. For images with **prediction theater curtain**, when feature[1063] < 0.9047, error rate increases to 0.2482 (**+14.33%**).
Figure 40: Visualization of feature[943]. For images with prediction forklift, when feature[943] < 1.1721, error rate increases to 0.2379 (+13.32%).

Figure 41: Visualization of feature[404]. For images with prediction french bulldog, when feature[404] < 0.2946, error rate increases to 0.2103 (+10.81%).
Figure 42: Visualization of feature [638]. For images with prediction syringe, when feature [638] < 0.2325, error rate increases to 0.3519 (+14.99%).

Figure 43: Visualization of feature [1634]. For images with prediction rhodesian ridgeback, when feature [1634] < 1.3779, error rate increases to 0.3184 (+10.91%).
Figure 44: Visualization of feature[1864]. For images with "tiger cat" label, when feature[1864] < 0.4673, error rate increases to 0.8786 (+10.71%).

| Class name     | Feature index | Decision rule | BER   | ER   | EC    | ALER  | Feature visualization | Feature name (from visualization) |
|----------------|---------------|---------------|-------|------|-------|-------|------------------------|----------------------------------|
| tiger cat      | 1864          | < 0.4673      | 0.7715 | 0.8786 | 0.9521 | 0.8473 | Figure 44              | green background                 |
| lighter        | 380           | < 1.2961      | 0.7285 | 0.8608 | 0.9335 | 0.8188 | Figure 45              | flame                            |
| purse          | 486           | < 0.1915      | 0.7277 | 0.9258 | 0.5011 | 0.7627 | Figure 46              | strings                          |
| chihuahua      | 198           | < 0.7873      | 0.7269 | 0.9332 | 0.7386 | 0.8062 | Figure 47              | close-up face                    |
| rifle          | 522           | < 1.4414      | 0.7223 | 0.9558 | 0.5985 | 0.7846 | Figure 48              | trigger                          |
| crayfish       | 1729          | < 1.0135      | 0.7154 | 0.9294 | 0.5806 | 0.7671 | Figure 49              | red fish skeleton                |
| cougar         | 1469          | < 0.6376      | 0.3438 | 0.6074 | 0.9172 | 0.5620 | Figure 50              | cougar nose                      |
| butternut squash| 1905         | < 0.6324      | 0.3438 | 0.6034 | 0.7830 | 0.5017 | Figure 51              | orange round edge                |
| sea cucumber   | 1147          | < 1.067       | 0.3438 | 0.6042 | 0.7718 | 0.4983 | Figure 52              | green round shape                |
| zucchini       | 752           | < 1.0602      | 0.3431 | 0.6445 | 0.8049 | 0.5416 | Figure 53              | green pipe                       |
| table lamp     | 1740          | < 2.5241      | 0.3423 | 0.7251 | 0.7528 | 0.5783 | Figure 54              | horizontal edge at bottom of table lamp |

Table 4: Results on a robust Resnet-50 model using grouping by label.
Figure 45: Visualization of feature[380]. For images with label lighter, when feature[380] < 1.2961, error rate increases to 0.8608 (+13.23%).

Figure 46: Visualization of feature[486]. For images with label purse, when feature[486] < 0.1915, error rate increases to 0.9258 (+19.81%).
Figure 47: Visualization of feature[198]. For images with label chihuahua, when feature[198] < 0.7873, error rate increases to 0.9332 (+20.63%).

Figure 48: Visualization of feature[522]. For images with label rifle, when feature[522] < 1.4414, error rate increases to 0.9558 (+23.35%).
Figure 49: Visualization of feature[1729]. For images with label crayfish, when feature[1729] < 1.0135, error rate increases to 0.9294 (+21.40%).

Figure 50: Visualization of feature[1469]. For images with label cougar, when feature[1469] < 0.6376, error rate increases to 0.6074 (+26.36%).
Figure 51: Visualization of feature[1905]. For images with label butternut squash, when feature[1905] < 0.6324, error rate increases to 0.6034 (+25.96%).

Figure 52: Visualization of feature[1147]. For images with label sea cucumber, when feature[1147] < 1.067, error rate increases to 0.6042 (+26.04%).
Figure 53: Visualization of feature[752]. For images with label zucchini, when feature[752] < 1.0602, error rate increases to 0.6445 (+30.14%).

Figure 54: Visualization of feature[752]. For images with label table lamp, when feature[1740] < 2.5241, error rate increases to 0.7251 (+38.28%).
Figure 55: Visualization of feature[1412]. For images with prediction paper towel, when feature[1412] < 1.2665, error rate increases to 0.7825 (+15.15%).

| Class name   | Feature index | Decision rule | BER    | ER     | EC     | ALER   | Feature visualization | Feature name (from visualization) |
|--------------|---------------|---------------|--------|--------|--------|--------|------------------------|----------------------------------|
| paper towel  | 1412          | < 1.2665      | 0.6310 | 0.7825 | 0.6566 | 0.6720 | Figure 55              | cylindrical hole                  |
| seat belt    | 1493          | < 1.0624      | 0.5983 | 0.7402 | 0.5816 | 0.6282 | Figure 56              | window                            |
| crutch       | 502           | < 0.7458      | 0.5842 | 0.7302 | 0.7233 | 0.6343 | Figure 57              | rods                              |
| lumbermill   | 56            | < 0.5817      | 0.5025 | 0.7049 | 0.4362 | 0.5817 | Figure 58              | tracks                            |
| bassoon      | 1104          | < 0.7026      | 0.5621 | 0.7490 | 0.6474 | 0.6208 | Figure 59              | hands and cylindrical bassoon     |
| impala       | 918           | < 0.9435      | 0.3535 | 0.6298 | 0.5917 | 0.4609 | Figure 60              | close-up face                     |
| boxer        | 404           | < 1.6458      | 0.3527 | 0.4991 | 0.7671 | 0.4246 | Figure 61              | dog nose                          |
| samoyed      | 1694          | < 0.7492      | 0.3487 | 0.5304 | 0.6247 | 0.4147 | Figure 62              | close-up dog face                 |
| milk can     | 676           | < 1.1286      | 0.3530 | 0.6284 | 0.6300 | 0.4707 | Figure 63              | horizontal edges                  |
| gasmask      | 835           | < 0.9034      | 0.3521 | 0.6216 | 0.5736 | 0.4514 | Figure 64              | round patches                     |
| king crab    | 952           | < 2.9012      | 0.3487 | 0.5991 | 0.5770 | 0.4396 | Figure 65              | crab tentacles                    |

Table 5: Results on a robust Resnet-50 model using grouping by prediction.
Figure 56: Visualization of feature [1493]. For images with prediction seat belt, when feature [1493] < 1.0624, error rate increases to 0.7402 (+14.19%).

Figure 57: Visualization of feature [502]. For images with prediction crutch, when feature [502] < 0.7458, error rate increases to 0.7302 (+14.60%).
Figure 58: Visualization of feature[56]. For images with prediction lumberhill, when feature[56] < 0.5817, error rate increases to 0.7049 (+14.24%).

Figure 59: Visualization of feature[1104]. For images with prediction bassoon, when feature[1104] < 0.7026, error rate increases to 0.7490 (+18.49%).
Figure 60: Visualization of feature[918]. For images with prediction impala, when feature[918] < 0.9435, error rate increases to 0.6298 (+27.63%).

Figure 61: Visualization of feature[404]. For images with prediction boxer, when feature[404] < 1.6458, error rate increases to 0.4991 (+14.64%).
Figure 62: Visualization of feature [1694]. For images with prediction samoyed, when feature [1694] < 0.7492, error rate increases to 0.5304 (+18.17%).

Figure 63: Visualization of feature [676]. For images with prediction milk can, when feature [676] < 1.1286, error rate increases to 0.6284 (+27.54%).
Figure 64: Visualization of feature[835]. For images with prediction gasmask, when feature[835] < 0.9034, error rate increases to 0.6216 (+26.95%).

Figure 65: Visualization of feature[952]. For images with prediction king crab, when feature[952] < 2.9012, error rate increases to 0.5991 (+25.04%).
G. Examples from Crowd study

The questionnaire for the Crowd study is in Figure 66.

G.1. Easy examples

Results are in Table 6.

| Class name                      | Class index | Feature index | Grouping | Feature visualization |
|---------------------------------|-------------|---------------|----------|-----------------------|
| malinois                        | 225         | 813           | prediction | Figure 67             |
| greenhouse, nursery, glasshouse | 580         | 1933          | prediction | Figure 68             |
| black and gold garden spider, Argiope aurantia | 72         | 652           | prediction | Figure 69             |
| scuba diver                     | 983         | 1588          | prediction | Figure 70             |
| sea cucumber, holothurian       | 329         | 28            | prediction | Figure 71             |

Table 6: Examples from the Amazon Mechanical Turk study that workers found as easy to describe.
Figure 67: Visualization of feature[813], class[225] and prediction grouping. Example descriptions: black fur, Canid eyes, facial fur, black and white, head region

Figure 68: Visualization of feature[1933], class[580] and prediction grouping. Example descriptions: plant, colorful flowers, leafy greens, bunch of plants, plant

Figure 69: Visualization of feature[652], class[72] and prediction grouping. Example descriptions: branching forms, shoes, body of creature, exotic arachnid, black color
Figure 70: Visualization of feature[1588], class[983] and prediction grouping. Example descriptions: tube or human, glowing faces, black, monkey-like, square face

Figure 71: Visualization of feature[28], class[329] and prediction grouping. Example descriptions: spots, rainbow, tubular sea creature, Tube, Tubular organism, belly
Figure 72: Visualization of feature[691], class[2] and prediction grouping. Example descriptions: structure, high contrast lines, Psychedelic colors, triangle

Figure 73: Visualization of feature[1211], class[125] and label grouping. Example descriptions: creature body, Shells, protruded or snug-fitting, video game, hard shell

G.2. Difficult examples
Results are in Table 7.

| Class name                                                                 | Class index | Feature index | Grouping | Feature visualization |
|---------------------------------------------------------------------------|-------------|---------------|----------|----------------------|
| great white shark, white shark, man-eater, man-eating shark, Carcharodon carcharias | 2           | 691           | prediction | Figure 72            |
| hermit crab                                                               | 125         | 1211          | label    | Figure 73            |
| goldfinch, Carduelis carduelis                                            | 11          | 788           | label    | Figure 74            |
| rock beauty, Holocanthus tricolor                                         | 392         | 1348          | label    | Figure 75            |
| pole                                                                      | 733         | 1107          | label    | Figure 76            |

Table 7: Examples from the Amazon Mechanical Turk study that workers found as difficult to describe.
Figure 74: Visualization of feature[788], class[11] and label grouping. Example descriptions: flying yellow being, rock, yellow spot, circular feathered body

Figure 75: Visualization of feature[1348], class[392] and label grouping. Example descriptions: edge, cave, nan, arrow shaped, rectangle

Figure 76: Visualization of feature[1107], class[733] and label grouping. Descriptions: long wooden beam, cube shapes, cells, rainbow hued circle, long pillars
H. User study with ML practitioners

| Role          | Participants                |
|---------------|----------------------------|
| ML Engineer   | [P2, P4, P5, P11, P18]     |
| Applied Scientist | [P9, P12]     |
| Researcher    | [P1, P7, P16, P17]        |
| Data Scientist | [P10, P20, P21]         |

| Experience in ML | Participants |
|------------------|--------------|
| 1 - 2 years      | 1 [P2]       |
| 2 - 5 years      | 4 [P5, P10, P11, P20] |
| 5 - 10 years     | 5 [P4, P7, P16, P17, P18] |
| > 10 years       | 4 [P1, P9, P12, P21]  |

Table 8: Distribution of roles and years of experience in Machine Learning among ML practitioners in the study.

| Class id | Class name       | Grouping | Robust Resnet-50 Top-1 Error | Participants     |
|----------|------------------|----------|-------------------------------|------------------|
| 424      | Barbershop       | prediction | 68.32%                        | 3 [P10, P18, P20] |
| 703      | Park Bench       | label     | 33.31%                        | 3 [P9, P11, P17] |
| 785      | Seat Belt        | label     | 33.23%                        | 4 [P2, P4, P12, P21] |
| 820      | Steam Locomotive | label     | 6.69%                         | 1 [P1]           |
| 282      | Tiger Cat        | label     | 77.15%                        | 3 [P5, P7, P16]  |

Table 9: Distribution of the first class groupings among machine-learning practitioners. The five examples contained features that were considered as “easy to describe” by Mturk workers to facilitate onboarding. The second class grouping was instead assigned randomly from the set of 120 class groupings that were part of the MTurk study.