Appendix: Pervasive Label Errors in Test Sets Destabilize Machine Learning Benchmarks

A Datasets

For our study, we select 10 of the most-cited, open-source datasets created in the last 20 years from the Wikipedia List of ML Research Datasets [26], with preference for diversity across computer vision, NLP, sentiment analysis, and audio modalities. Citation counts were obtained via the Microsoft Cognitive API. In total, we evaluate six visual datasets: MNIST, CIFAR-10, CIFAR-100, Caltech-256, ImageNet, and QuickDraw; three text datasets: 20news, IMDB, and Amazon Reviews; and one audio dataset: AudioSet.

A.1 Dataset details

For each of the datasets we investigate, we summarize the original data collection and labeling procedure as they pertain to potential label errors.

MNIST [22]. MNIST is a database of binary images of handwritten digits. The dataset was constructed from Handwriting Sample Forms distributed to Census Bureau employees and high school students; the ground-truth labels were determined by matching digits to the instructions of the task to copy a particular set of digits [11]. Label errors may arise from failure to follow instructions or from handwriting ambiguities.

CIFAR-10 / CIFAR-100 [21]. The CIFAR-10 and CIFAR-100 datasets are collections of small 32×32 images and labels from a set of 10 or 100 classes, respectively. The images were collected by searching the internet for the class label. Human labelers were instructed to select images that matched their class label (query term) by filtering out mislabeled images. Images were intended to only have one prominent instance of the object, but could be partially occluded as long as it was identifiable to the labeler.

Caltech-256 [10]. Caltech-256 is a database of images sorted into 256 classes, plus an extra class called "clutter". Images were scraped from image search engines. Four human labelers were instructed to rate the images into "good," "bad," and "not applicable," eliminating the images that were confusing, occluded, cluttered, artistic, or not an example of the object category from the dataset. Because no explicit test set is provided, we study label errors in the entire dataset to ensure coverage of any test set split used by practitioners. **Modifications**: In our study, we ignore data with the ambiguous "clutter" label (class 257) and consider only the images labeled class 1 to class 256.

ImageNet [6]. ImageNet is a database of images belonging to one of 1,000 classes. Images were scraped by querying words from WordNet "synonym sets" (synsets) on several image search engines. The images were labeled by Amazon Mechanical Turk workers who were asked whether each image contains objects of a particular given synset. Workers were instructed to select images that contain objects of a given subset regardless of occlusions, number of objects, and clutter to "ensure diversity" in the dataset's images.

QuickDraw [12]. The Quick, Draw! dataset contains more than 1 billion doodles collected from users of an experimental game to benchmark image classification models. Users were instructed to draw pictures corresponding to a given label, but the drawings may be "incomplete or may not match the label." Because no explicit test set is provided, we study label errors in the entire dataset to ensure coverage of any test set split used by practitioners.

20news [30]. The 20 Newsgroups dataset is a collection of articles posted to Usenet newsgroups used to benchmark text classification and clustering models. The label for each example is the newsgroup it was originally posted in (e.g. "misc.forsale"), so it is obtained during the overall data collection procedure.

IMDB [27]. The IMDB Large Movie Review Dataset is a collection of movie reviews to benchmark binary sentiment classification. The labels were determined by the user's review: a score ≤ 4 out of 10 is considered negative; ≥ 7 out of 10 is considered positive.

Amazon Reviews [29]. The Amazon Reviews dataset is a collection of textual reviews and 5-star ratings from Amazon customers used to benchmark sentiment analysis models. We use the 5-core (9.9 GB) variant of the dataset. **Modifications**: In our study, 2-star and 4-star reviews are removed due to ambiguity with 1-star and 5-star reviews, respectively. If these reviews were left in the dataset, they could inflate error counts. Because no explicit test set is provided, we study label errors in the entire dataset to ensure coverage of any test set split used by practitioners.

AudioSet [8]. AudioSet is a collection of 10-second sound clips drawn from YouTube videos and multiple labels describing the sounds that are present in the clip. Three human labelers independently rated the presence of one or more labels (as "present," "not present," and "unsure"), and majority agreement was required to assign a label. The authors note that spot checking revealed some label errors due to "confusing labels, human error, and difference in detection of faint/non-salient audio events."

B Mechanical Turk details

Mechanical Turk budget Mechanical Turk workers were paid an hourly rate of $7.20 (based on an estimated evaluation time of 5 seconds per image). In total, we spent $1623.29 on human verification experiments on Mechanical Turk. Results would likely improve with a larger budget.

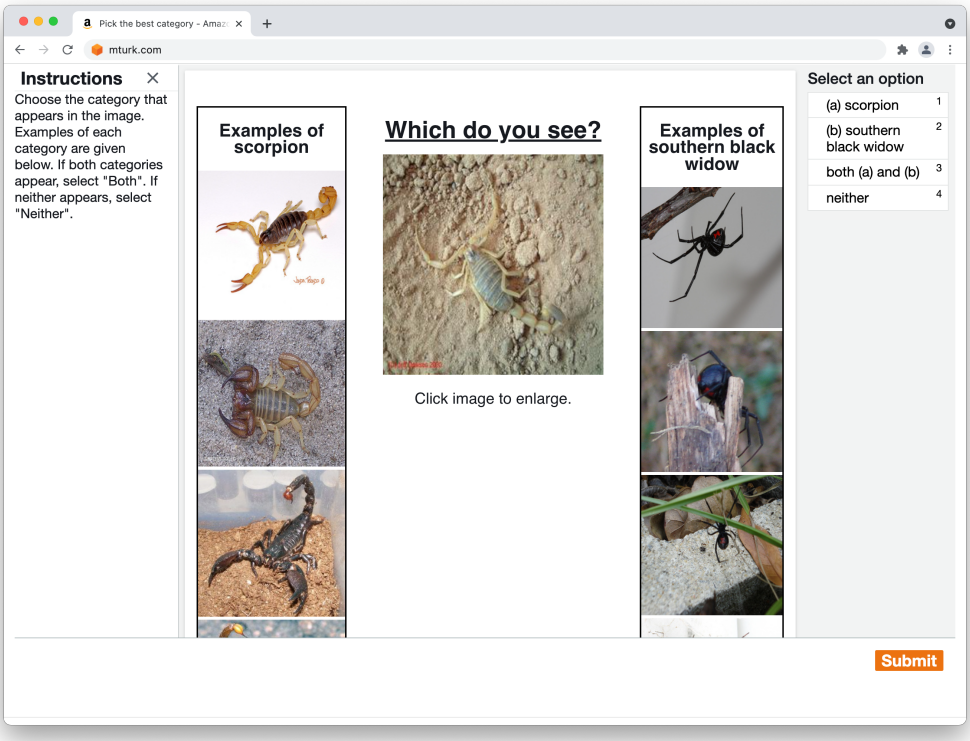


Figure S1: Mechanical Turk worker interface showing an example from ImageNet (with given label "southern black window"). For each data point algorithmically identified as a potential label error, the interface presents the data point, along with examples belonging to the given class. The interface also shows data points belonging to the confidently predicted class (in this case, "scorpion"). Either the given label is shown as option (a) and the predicted label is shown as option (b), or vice versa (chosen randomly). The worker is asked whether the image belongs to class (a), (b), both, or neither.

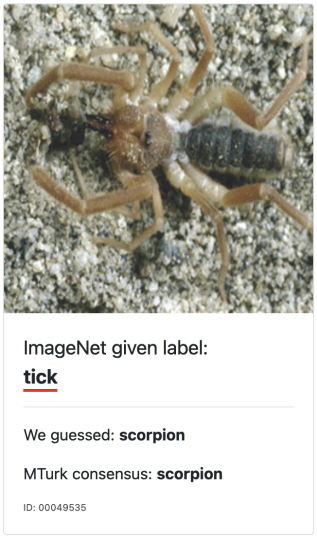


Figure S2: An example from https://labelerrors.com that Mechanical Turk workers got wrong. The image clearly doesn't match the ImageNet given label "tick," but upon close inspection, it does not match the predicted label "scorpion" either. The insect shown is in fact an arachnid of the order Solifugae, commonly known as camel spiders or wind scorpions. Despite the common name, this animal is not a true scorpion.

C Details of confident learning (CL) for finding label errors

Here we summarize CL joint estimation and how it is used to algorithmically flag candidates with likely label errors for subsequent human review. An unnormalized representation of the joint distribution between observed and true label, called the *confident joint* and denoted $\boldsymbol{C}_{\tilde{y},y^*}$, is estimated by counting all the examples with noisy label $\tilde{y}=i$, with high probability of actually belonging to label $y^*=j$. This binning can be expressed as:

$$\boldsymbol{C}_{\tilde{y},y^*} = |\{\boldsymbol{x} \in \boldsymbol{X}_{\tilde{y}=i} : \ \hat{p}(\tilde{y}=j;\boldsymbol{x},\boldsymbol{\theta}) \geq t_j\}|$$

where $\boldsymbol{x}$ is a data example (e.g. an image), $\boldsymbol{X}_{\tilde{y}=i}$ is the set of examples with noisy label $\tilde{y}=i$, $\hat{p}(\tilde{y}=j;\boldsymbol{x},\boldsymbol{\theta})$ is the out-of-sample predicted probability that example $\boldsymbol{x}$ actually belongs to noisy class $\tilde{y}=j$ (even though its given label $\tilde{y}=i$) for a given model $\boldsymbol{\theta}$. Finally, t_j is a per-class threshold that, in comparison to other confusion matrix approaches, provides robustness to heterogeneity in class distributions and class distributions, defined as:

$$t_j = \frac{1}{|\boldsymbol{X}_{\tilde{y}=j}|} \sum_{\boldsymbol{x} \in \boldsymbol{X}_{\tilde{y}=j}} \hat{p}(\tilde{y}=j;\boldsymbol{x},\boldsymbol{\theta}) \tag{1}$$

A caveat occurs when an example is confidently counted into more than one bin. When this occurs, the example is only counted in the $\arg\max_{l \in [m]} \hat{p}(\tilde{y}=l;\boldsymbol{x},\boldsymbol{\theta})$ bin.

$\boldsymbol{Q}_{\tilde{y},y^*}$ is estimated by normalizing $\boldsymbol{C}_{\tilde{y},y^*}$, as follows:

$$\hat{\boldsymbol{Q}}_{\tilde{y}=i,y^*=j} = \frac{\frac{\boldsymbol{C}_{\tilde{y}=i,y^*=j}}{\sum_{j \in [m]} \boldsymbol{C}_{\tilde{y}=i,y^*=j}} \cdot |\boldsymbol{X}_{\tilde{y}=i}|}{\sum_{i \in [m], j \in [m]} \left(\frac{\boldsymbol{C}_{\tilde{y}=i,y^*=j}}{\sum_{j \in [m]} \boldsymbol{C}_{\tilde{y}=i,y^*=j}} \cdot |\boldsymbol{X}_{\tilde{y}=i}| \right)} \tag{2}$$

The numerator calibrates $\sum_j \hat{\boldsymbol{Q}}_{\tilde{y}=i,y^*=j} = |\boldsymbol{X}_i| / \sum_{i \in [m]} |\boldsymbol{X}_i|, \forall i \in [m]$ so that row-sums match the observed prior over noisy labels. The denominator makes the distribution sum to 1.

D Failure modes of confident learning

Confident learning can fail to exactly estimate $\boldsymbol{X}_{\tilde{y}=i,y^*=j}$ (the set of examples with noisy label i and actual label j) when either:

- **Case 1**: $\hat{p}(\tilde{y}{=}j;\boldsymbol{x},\boldsymbol{\theta}) < t_j \longrightarrow \boldsymbol{x} \notin \hat{\boldsymbol{X}}_{\tilde{y}=i,y^*=j}$, or
- **Case 2**: $\hat{p}(\tilde{y}{=}k;\boldsymbol{x},\boldsymbol{\theta}) \geq t_k \longrightarrow \boldsymbol{x} \in \hat{\boldsymbol{X}}_{\tilde{y}=i,y^*=k}$, for some $k \neq j$

where t_j is the per-class average threshold (Eqn. 1 above, in Appendix C). In the real-world datasets we study, the predicted probabilities are noisy such that $\hat{p}_{\boldsymbol{x},\tilde{y}=j} = p^*_{\boldsymbol{x},\tilde{y}=j} + \epsilon_{\boldsymbol{x},\tilde{y}=j}$, where $\hat{p}_{\boldsymbol{x},\tilde{y}=j}$ is shorthand for $\hat{p}(\tilde{y}{=}j;\boldsymbol{x},\boldsymbol{\theta})$; $p^*_{\boldsymbol{x},\tilde{y}=j}$ is the ideal/non-noisy predicted probability; and $\epsilon_{\boldsymbol{x},\tilde{y}=j} \in \mathcal{R}$ is the error/deviation from ideal. Unlike learning with perfect labels, $p^*_{\boldsymbol{x},\tilde{y}=j}$ is not always 0 or 1 because in our setting some classes are mislabeled as other classes some fraction of the time. Expressing the two failure cases in terms of error, we have:

- **Case 1**: $\epsilon_{\boldsymbol{x},\tilde{y}=j} < t_j - p^*_{\boldsymbol{x},\tilde{y}=j} \longrightarrow \boldsymbol{x} \notin \hat{\boldsymbol{X}}_{\tilde{y}=i,y^*=j}$, or
- **Case 2**: $\epsilon_{\boldsymbol{x},\tilde{y}=k} \geq t_k - p^*_{\boldsymbol{x},\tilde{y}=k} \longrightarrow \boldsymbol{x} \in \hat{\boldsymbol{X}}_{\tilde{y}=i,y^*=k}$, for some $k \neq j$

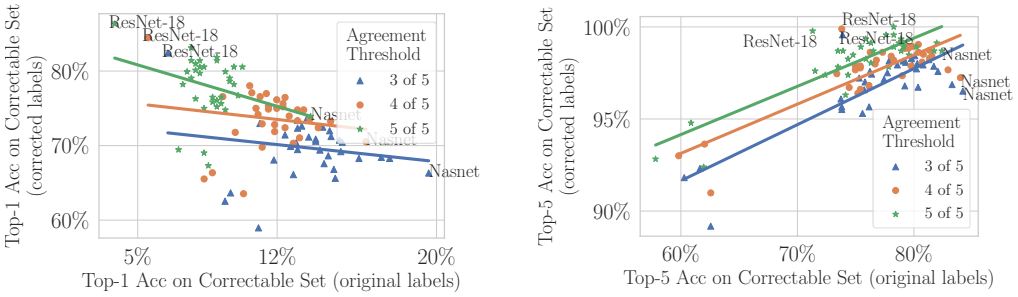
Case 1 bounds the error of $\hat{p}(\tilde{y}{=}j;\boldsymbol{x},\boldsymbol{\theta})$ (in the limit to $-\infty$) and Case 2 bound the error of $\hat{p}(\tilde{y}{=}j;\boldsymbol{x},\boldsymbol{\theta})$ (in the limit to ∞) such that when either occurs, $\exists (i,j) \in [m]\times[m]$, s.t. $\hat{\boldsymbol{X}}_{\tilde{y}=i,y^*=j} \neq \boldsymbol{X}_{\tilde{y}=i,y^*=j}$, i.e., we imperfectly estimate the label errors prior to human validation. Figure 2 shows uniquely challenging examples (with excessively erroneous $\hat{p}(\tilde{y}{=}j;\boldsymbol{x},\boldsymbol{\theta})$) when these failure mode cases potentially occur.

E Reproducibility and computational requirements

For all 10 datasets, label errors were found using a Linux 18.04 LTS server comprising 128GB of memory, an Intel Core i9-9820X Skylake X 10-Core 3.3GHz, and one RTX 2080 TI GPU. We open-source a single script to reproduce the label errors for every dataset at https://github.com/cleanlab/label-errors/blob/main/examples/Tutorial%20-%20How%20To%20Find%20Label%20Errors%20With%20CleanLab.ipynb. Reproducing the label errors for all 10 datasets using this tutorial takes about 5 minutes on a modern consumer-grade laptop (e.g., a 2021 Apple M1 MacBook Air).

F Additional findings on implications of label errors in test data

Here we provide some additional details/results to complement Section 5 from the main text. Figure 3 depicts how the benchmarking rankings on the correctable subset of ImageNet examples change significantly for an *agreement threshold* $= 5$, meaning 5 of 5 human raters need to independently select the same alternative label for that data point and a new label to be included in the accuracy evaluation. To ascertain that the results of this figure are not due to the setting of the agreement threshold, the results for all three settings of the agreement threshold are shown in Sub-figure S3b. Observe the negative correlation (for top-1 accuracy) occurs in all three settings. Furthermore, observe that this negative correlation no longer holds when top-5 accuracy is used (shown in S3a), likely because many of these models use a loss which maximizes (and overfits to noise) based on top-1 accuracy, not top-5 accuracy. Regardless of whether top-1 or top-5 accuracy is used, model benchmark rankings change significantly on the correctable set in comparison to the original test set (see Table S1).



(a) Top-1 Accuracy.

(b) Top-5 Accuracy.

Figure S3: Benchmark ranking comparison of 34 pre-trained models on the ImageNet val set (used as test data here) for various settings of the agreement threshold. Top-5 benchmarks are unchanged by removing label errors (a), but change drastically on the correctable subset with original (erroneous) labels versus corrected labels. Corrected test set sizes: 1428 (▲), 960 (●), 468 (⋆).

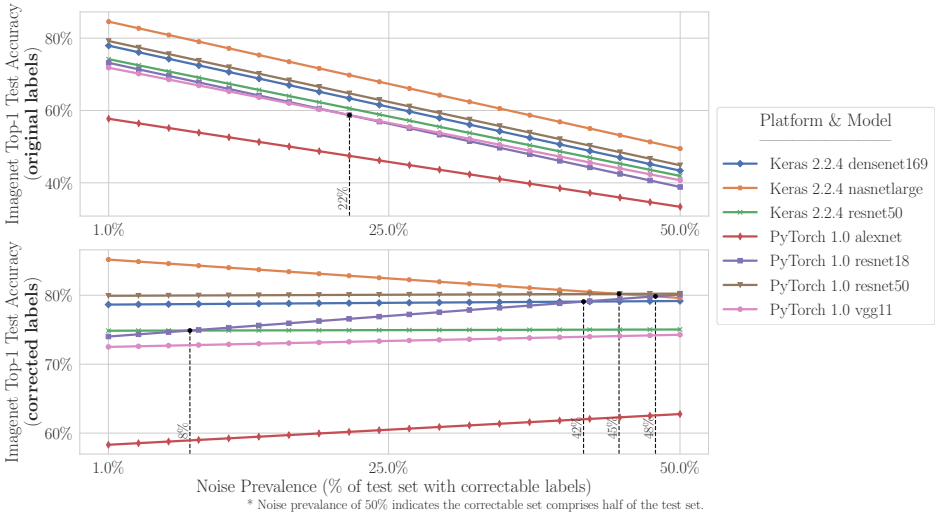


Figure S4: ImageNet top-1 original accuracy (top panel) and top-1 corrected accuracy (bottom panel) vs Noise Prevalence with agreement threshold = 5 (instead of threshold = 3, c.f., Figure 4).

The dramatic changes in ranking shown in Table S1 may be explained by overfitting to the validation set when these models are trained, which can occur inadvertently during hyper-parameter tuning, or by overfitting to the noise in the training set. These results also suggest that keeping some correct labels on a secret correctable set of label errors may provide a useful framework for detecting overfitting on test sets toward a more reliable approach for benchmarking generalization accuracy across ML models.

Table S1: Individual accuracy scores for Sub-figure 3b with *agreement threshold = 3 of 5*. Acc@1 stands for the (top-1 validation) original accuracy on the correctable set, in terms of original ImageNet examples and labels. *cAcc@1* stands for the (top-1 validation) corrected accuracy on the correctable set of ImageNet examples with correct labels. To be corrected, at least 3 of 5 Mechanical Turk raters had to independently agree on a new label, proposed by us using the class with the $\arg\max$ probability for the example.

Platform	Model	Acc@1	cAcc@1	Acc@5	cAcc@5	Rank@1	cRank@1	Rank@5	cRank@5
PyTorch 1.0	resnet18	6.51	82.42	73.81	99.58	34	1	30	1
PyTorch 1.0	resnet50	13.52	73.74	79.97	98.46	20	2	11	2
PyTorch 1.0	vgg19_bn	13.03	73.39	79.97	97.97	23	3	10	9
PyTorch 1.0	vgg11_bn	11.13	72.97	76.26	97.55	30	4	22	15
PyTorch 1.0	resnet34	13.24	72.62	77.80	98.11	21	5	18	6
PyTorch 1.0	densenet169	14.15	72.55	79.62	98.32	16	6	12	3
PyTorch 1.0	densenet121	14.29	72.48	78.64	97.97	14	7	16	11
PyTorch 1.0	vgg19	13.03	72.34	79.34	98.04	22	8	13	8
PyTorch 1.0	resnet101	14.64	71.99	81.16	98.25	11	9	5	4
PyTorch 1.0	vgg16	12.39	71.43	77.52	97.20	28	10	19	19
PyTorch 1.0	densenet201	14.71	71.22	80.81	97.97	10	11	6	10
PyTorch 1.0	vgg16_bn	13.59	71.15	77.87	97.41	19	12	17	17
Keras 2.2.4	densenet169	13.94	70.87	78.85	98.18	17	13	15	5
PyTorch 1.0	densenet161	15.13	70.73	80.11	98.04	7	14	8	7
Keras 2.2.4	densenet121	13.94	70.59	76.40	97.48	18	15	20	16
PyTorch 1.0	resnet152	15.27	70.45	81.79	97.83	5	16	4	12
PyTorch 1.0	vgg11	12.96	70.38	75.49	97.27	25	17	27	18
PyTorch 1.0	vgg13_bn	12.68	69.89	75.84	96.99	27	18	25	20
PyTorch 1.0	vgg13	13.03	69.47	76.40	96.78	24	19	21	24
Keras 2.2.4	nasnetmobile	14.15	69.40	79.27	96.85	15	20	14	21
Keras 2.2.4	densenet201	15.20	69.19	80.11	97.76	6	21	9	13
Keras 2.2.4	mobilenetV2	14.57	68.63	75.84	96.57	12	22	24	26
Keras 2.2.4	inceptionresnetv2	17.23	68.42	83.40	96.85	3	23	2	22
Keras 2.2.4	xception	17.65	68.28	82.07	97.62	2	24	3	14
Keras 2.2.4	inceptionv3	16.11	68.28	80.25	96.78	4	25	7	23
Keras 2.2.4	vgg19	11.83	68.07	73.95	95.52	29	26	29	30
Keras 2.2.4	mobilenet	14.36	67.58	73.60	96.08	13	27	31	27
Keras 2.2.4	resnet50	14.85	66.81	76.12	95.73	9	28	23	28
Keras 2.2.4	nasnetlarge	19.61	66.32	84.24	96.57	1	29	1	25
Keras 2.2.4	vgg16	12.82	66.11	74.09	95.66	26	30	28	29
PyTorch 1.0	inception_v3	14.92	65.62	75.56	95.38	8	31	26	31
PyTorch 1.0	squeezenet1_0	9.66	63.66	60.50	91.88	32	32	34	33
PyTorch 1.0	squeezenet1_1	9.38	62.54	61.97	92.30	33	33	33	32
PyTorch 1.0	alexnet	11.06	58.96	62.61	89.29	31	34	32	34

The benchmarking experiment was replicated on CIFAR-10 in addition to ImageNet. The individual accuracies for CIFAR-10 are reported in Table S2. Similar to ImageNet, lower capacity models tend to outperform higher capacity models when benchmarked using corrected labels (instead of the original, erroneous labels).

Whereas traditional notions of benchmarking generalization accuracy assume the train and test distributions are the same, this is nonsensical in the case of noisy training data — the test dataset should never contain noise because in real-world applications, we want a trained model to predict the error-free outputs on unseen examples, and benchmarking should measure as such. In two independent experiments in ImageNet and CIFAR-10, we observe that models, pre-trained on the original (noisy) datasets, with less expressibility (e.g., ResNet-18) tend to outperform higher capacity models (e.g., NASNet) on the corrected test set labels.

Table S2: Individual CIFAR-10 accuracy scores for Sub-figure 3c with *agreement threshold = 3 of 5*. Acc@1 stands for the top-1 validation accuracy on the correctable set ($n = 18$) of original CIFAR-10 examples and labels. See Table S1 caption for more details. Discretization of accuracies occurs due to the limited number of corrected examples on the CIFAR-10 test set.

Platform	Model	Acc@1	cAcc@1	Acc@5	cAcc@5	Rank@1	cRank@1	Rank@5	cRank@5
PyTorch 1.0	googlenet	55.56	38.89	94.44	94.44	1	10	13	13
PyTorch 1.0	vgg19_bn	50.00	38.89	100.00	100.00	2	11	7	7
PyTorch 1.0	densenet169	44.44	50.00	100.00	100.00	5	4	2	2
PyTorch 1.0	vgg16_bn	44.44	44.44	100.00	100.00	3	8	5	5
PyTorch 1.0	inception_v3	44.44	33.33	100.00	100.00	6	12	8	8
PyTorch 1.0	resnet18	44.44	55.56	94.44	100.00	4	2	10	10
PyTorch 1.0	densenet121	38.89	50.00	100.00	100.00	8	5	3	3
PyTorch 1.0	densenet161	38.89	50.00	100.00	100.00	9	6	4	4
PyTorch 1.0	resnet50	38.89	44.44	100.00	100.00	7	9	6	6
PyTorch 1.0	mobilenet_v2	38.89	27.78	100.00	100.00	10	13	9	9
PyTorch 1.0	vgg11_bn	27.78	66.67	100.00	100.00	11	1	1	1
PyTorch 1.0	resnet34	27.78	55.56	94.44	100.00	13	3	11	11
PyTorch 1.0	vgg13_bn	27.78	50.00	94.44	100.00	12	7	12	12

G Expert label review details

To mitigate possible bias in our expert reviewing process, we did not show reviewers whether a particular image was CL-flagged or not, and we randomized whether a CL-flagged or non CL-flagged image was shown first for each ImageNet class. We also randomized whether the given or predicted label was the first or second choice offered to the reviewer. We did not however randomize the class order as reviewing was much more efficient when the classes were presented in order (required less drastic context switching) and helped reviewers to learn while reviewing, especially for taxonomies with many related classes (e.g., dog breeds). The three authors of this paper, aided by an experienced data labeler, served as these expert reviewers, spending around 67 seconds in total on average to review each image label (14x more time than MTurk workers) and around 109 seconds on average to review the images where a second phase was required for the expert reviewers to come to consensus due to disagreement (28x more time than MTurk workers).

There were 66 ImageNet classes (out of the 1000) that had no CL-flagged image in the validation set. For these classes, the experts could not review a CL-flagged image, but experts still reviewed a non CL-flagged image. Thus, 1934 images were reviewed by experts (934 CL-flagged and 1000 non-CL flagged). These images were assigned into 3 non-disjoint evenly-sized partitions (one for each expert to review) such that each image was reviewed by at least 2 experts. Expert reviewer 1 was assigned images from classes 1-666. Expert reviewer 2 was assigned classes 1-333 and 667-1000. Expert reviewer 3 was assigned classes 334-1000. After independently reviewing the images (spending 54 seconds per image, on average), experts disagreed on 438 images. The experts subsequently discussed each of these images to reach a consensus decision (spending 55 seconds on average in discussions to come to consensus on a choice for each label). Table S3 counts the different types of label issues identified by experts in the CL-flagged and non-CL flagged images, from which we computed the percentages reported in Table 3.

The time spent for expert review in Table 3 is computed as: (1934 / 1934) * 54 seconds + (438 / 1934) * 55 seconds = 67 seconds (i.e., time spent on average for all 1934 images for independent expert review + additional time spent on the 438 images requiring experts to discuss their choices and come to agreement).

In some cases, experts agreed that neither the given nor the predicted label was appropriate, but Mechanical Turk workers chose the predicted label. These were tricky cases which often required careful scrutiny to identify the true class of the given image. Figure S2 shows an example of such a case, where the image clearly doesn't match the ImageNet given label, and upon close inspection, doesn't match the predicted label either.

Table S3: Counts of various types of label issues identified by experts in CL-flagged examples vs non-CL flagged examples from ImageNet (see Section 6). Here, count(errors) = count(correctable) + count(multi-label) + count(neither) + count(non-agreement). Also, count(total) = count(non-errors) + count(errors). After independently making decisions about each label, experts were subsequently required to resolve any non-agreement by reaching a consensus via group deliberation. There were 66 ImageNet classes which did not have a CL-flagged error, thus only 934 CL-flagged examples were reviewed instead of 1000 (1 example for every class).

	total	non-errors	errors	correctable	multi-label	neither	non-agreement
CL (MTurk)	934	481	453	205	92	53	103
CL (expert)	934	548	386	165	122	99	0
non-CL (expert)	1000	840	160	32	91	37	0