Pervasive Label Errors in ML Benchmark Test Sets, Consequences, and Benefits

Curtis G. Northcutt
MIT
Cambridge, MA 02115
cgn@mit.edu

Anish Athalye
MIT
Cambridge, MA 02115
aathalye@mit.edu

Jessy Lin
University of California Berkeley
Berkeley, CA 94720
jessy_lin@berkeley.edu

Abstract

We use new algorithmic techniques to automatically identify numerous label errors in the test sets of ten of the most commonly-used computer vision, natural language, and audio datasets. Errors are widespread: validated using human studies, we estimate an average of 3.4% errors across ten datasets, where for example 2916 errors comprise 6% of the ImageNet validation set. In a case study on ImageNet, we find that label errors do not corrupt current benchmarks. Unexpectedly, we find a use for erroneously labeled test data: as a "honeypot" for reliable benchmarking of generalization accuracy. In ImageNet, pre-trained classifiers exhibit a negative correlation in performance on corrected labels versus performance on original (erroneous) labels on the validation set, suggesting that this honeypot technique may be effective as a tool for measuring overfitting to the validation set.

1 Introduction

Large, labeled data sets have been critical to the success of supervised machine learning across the board in domains such as image classification, sentiment analysis, and audio classification. Labeled data is usually equated with ground truth. However, this is a fallacy, because datasets can have errors. The processes used to construct datasets often involve some degree of automatic labeling or crowd-sourcing, techniques that are inherently error-prone. Even with controls for error correction [14, 35], errors can slip through. Prior work has considered the consequences of noisy labels, usually in the context of *learning* with noisy labels in the train set, and with algorithmic findings validated through experiments with synthetic label noise. Past research has concluded that label noise is not a major concern, because of techniques to learn with noisy labels [25, 22], and also because deep learning is believed to be naturally robust to label noise [28, 30, 12, 19].

However, label errors in *test* sets have a different set of potential consequences. Researchers rely on benchmark test datasets to evaluate and measure progress in the state-of-the-art and to validate theoretical findings. If label errors occurred profusely, they could potentially undermine the framework by which we measure progress in machine learning.

We present the first study that systematically analyzes the prevalence of label errors across ten commonly-used datasets across computer vision, natural language processing, and audio processing. Unlike prior work on noisy labels, we do not experiment with synthetic noise but with naturally-occurring errors. Using *confident learning* [23], we identify label errors in test sets at scale, and validate these label errors through human evaluation, estimating over five million (10%) errors in one dataset. Figure 1 shows examples of validated label errors.

We use ImageNet as a case study to understand the consequences of label errors in test sets. While there are a large number (2916, over 6%) of erroneous labels in the ImageNet validation set, we find that benchmarking is currently not affected by the presence of these label errors.

34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada.

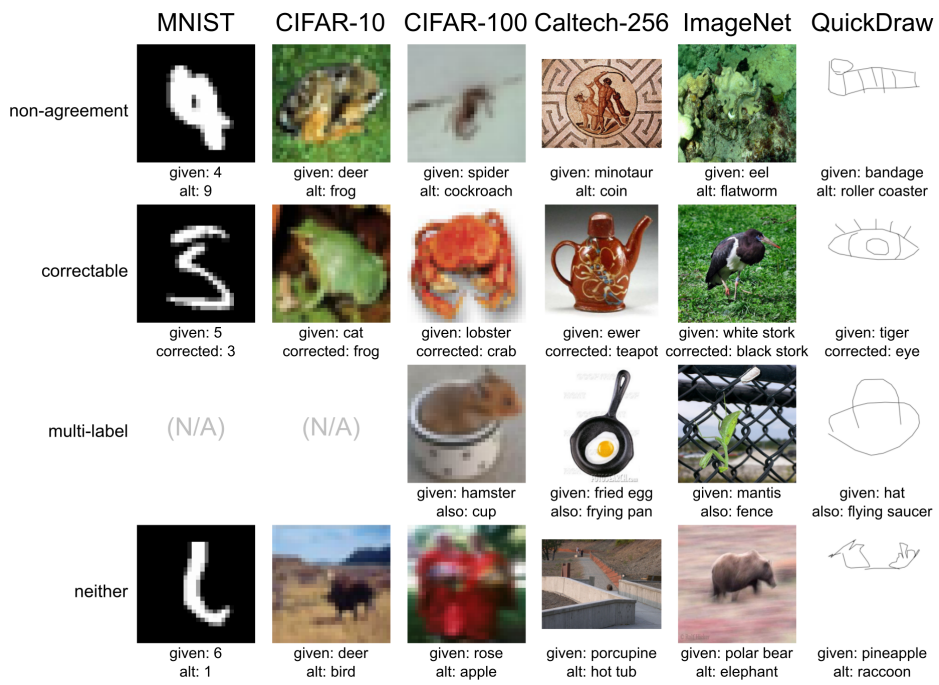


Figure 1: Example label errors and their categorization for image datasets.

We find an unexpected use for label errors in the validation set: as a "honeypot" for reliable benchmarking of model generalization accuracy. When benchmarking ImageNet classifiers on what we call the honeypot set, validation data that was confirmed to be labeled incorrectly, we find a negative correlation between rankings on corrected labels versus the original (erroneous) labels, suggesting that models may be overfitting to the validation set. We hypothesise that this honeypot technique may be generally useful for reliable benchmarking in these datasets as a measure for overfitting to test sets.

To our knowledge, our work is the first demonstration of learning with noisy labels on real (non-synthetic) datasets at scale across many datasets, modalities, distributions, and amounts of data.

2 Related work

Finding label errors is related to learning with noisy labels [24, 32, 22, 13, 29], but these approaches modify the loss without explicit error identification, or require model-specific approaches that do not generalize across ten diverse datasets. [34, 11, 2, 17] use confusion matrices for noise quantification error finding, but lack either robustness to class imbalance or theoretical support. Confident learning (CL) [23] assembles these approaches into a model-agnostic framework with robustness to class imbalance and theoretical support for exactly finding of label errors; hence, our choice to use CL. Crowd-sourced curation of labels via multiple workers [35, 4, 26] is used to validate these errors. [31] confirm multi-class label issues in the ImageNet dataset; we expand label error finding to more types of errors and datasets.

3 Dataset selection

We consider ten of the most-cited, open-source datasets created in the last 20 years from the Wikipedia List of ML Research Datasets [3], with preference for diversity across computer vision, NLP, sentiment analysis, and audio modalities. Citations were obtained via the Microsoft Cognitive API. Details about each dataset, the original collection procedure of the labels in each dataset, and any modifications we made (e.g. removing 2-star and 4-star classes from Amazon Reviews) are available in the appendices (Sec. B). In total, we evaluate six visual datasets: MNIST [16], ImageNet [5], QuickDraw [10], CIFAR-10, CIFAR-100 [15], and Caltech-256 [8]; three text datasets: IMDB [18], Amazon Reviews [20], and 20news [21]; and one audio dataset: AudioSet [6].

4 Identifying label errors in benchmark datasets

To identify label errors in a test dataset with n examples and m classes, we first characterize label noise in the dataset using the confident learning (CL) framework [23] to estimate $\boldsymbol{Q}_{\tilde{y},y^*}$, the $m \times m$ discrete joint distribution of observed, noisy labels, $\tilde{y}$, and unknown, true labels, y^*. Inherent in $\boldsymbol{Q}_{\tilde{y},y^*}$ is the assumption that noise is class-conditional [1], depending only on the latent true class, not the data. This assumption is commonly used [7, 29] because it is reasonable. For example, in ImageNet, a *tiger* is more likely to be mislabeled *cheetah* than *flute*.

The diagonal entry, $\hat{p}(\tilde{y}{=}i, y^*{=}i)$, of matrix $\boldsymbol{Q}_{\tilde{y},y^*}$ is the probability that examples in class i are correctly labeled. Thus, if the dataset is error-free, then $\sum_{i\in[m]} \hat{p}(\tilde{y}{=}i, y^*{=}i) = 1$. The fraction of label errors is $\rho = 1 - \sum_{i\in[m]} \hat{p}(\tilde{y}{=}i, y^*{=}i)$ and the number of label errors is $\rho \cdot n$. To find label errors, we choose the top $\rho \cdot n$ examples ordered by the normalized margin: $\hat{p}(\tilde{y}{=}i; \boldsymbol{x}, \boldsymbol{\theta}) - \max_{j\neq i} \hat{p}(\tilde{y}{=}j; \boldsymbol{x}, \boldsymbol{\theta})$ [33]. Table 1 shows the number of CL guessed label errors for each test set.

CL $\boldsymbol{Q}_{\tilde{y},y^*}$ estimation is summarized in the appendices (see Sec. A). To produce the results in this paper, we used the open-source implementation of confident learning in the `cleanlab` package[1].

Computing out-of-sample predicted probabilities Estimating $\boldsymbol{Q}_{\tilde{y},y^*}$ for CL noise characterization requires two inputs for each dataset: (1) out-of-sample predicted probabilities $\hat{\boldsymbol{P}}_{k,i}$ ($n \times m$ matrix) and (2) the test set labels $\tilde{y}_k$. We observe best results computing $\hat{\boldsymbol{P}}_{k,i}$ by pre-training on the train set, then fine-tuning (all layers) on the test set using cross-validation to ensure $\hat{\boldsymbol{P}}_{k,i}$ is out-of-sample. If pre-trained models are easily accessible and open-sourced (e.g. ImageNet), we use them instead of pre-training ourselves. If the dataset did not have an explicit test set (e.g. QuickDraw), we skip pre-training, and compute $\hat{\boldsymbol{P}}_{k,i}$ using cross-validation on the entire dataset. For all datasets, we try multiple models that achieve near state-of-the-art accuracy with light hyper-parameter tuning, and use the model yielding highest cross-validation accuracy, reported in Table 1. Code to reproduce the results in Table 1 is open-sourced at [[obscured for blind-review]].

Using this approach allows us to find label errors without manually checking the entire test set, because CL identifies potential label errors automatically.

Table 1: Test set errors prominent across common benchmark datasets. Errors are estimated using confident learning (CL) and validated by human workers on Mechanical Turk.

Dataset	Modality	Size	Model	Test Set Errors				
				CL guessed	MTurk checked	validated	estimated	% error
MNIST	image	10,000	2-conv CNN	100	100 (100%)	15	-	0.15
CIFAR-10	image	10,000	VGG	275	275 (100%)	54	-	0.54
CIFAR-100	image	10,000	VGG	2235	2235 (100%)	585	-	5.85
Caltech-256	image	30,607	ResNet-152	4,643	400 (8.6%)	65	754	2.46
ImageNet*	image	50,000	ResNet-50	5,440	5,440 (100%)	2,916	-	5.83
QuickDraw	image	50,426,266	VGG	6,825,383	2,500 (0.04%)	1870	5,105,386	10.12
20news	text	7,532	TFIDF + SGD	93	93 (100%)	82	-	1.11
IMDB	text	25,000	FastText	1,310	1,310 (100%)	725	-	2.9
Amazon	text	9,996,437	FastText	533,249	1,000 (0.2%)	732	390,338	3.9
AudioSet	audio	20,371	VGG	307	307 (100%)	275	-	1.35

*Because the ImageNet test set labels are not publicly available, the ILSVRC 2012 validation set is used.

5 Validating label errors

We validated algorithmically-identified label errors with a Mechanical Turk study. For three datasets with a large number of errors, we checked a random sample; for the rest, we checked all errors.

We presented workers with hypothesized errors and asked them whether they saw the (1) given label, (2) the top CL-predicted label, (3) both labels, or (4) neither label in the example. To aid the worker, the interface included examples, drawn from the training set, of the given class and the CL-predicted class. See Appendix C for details.

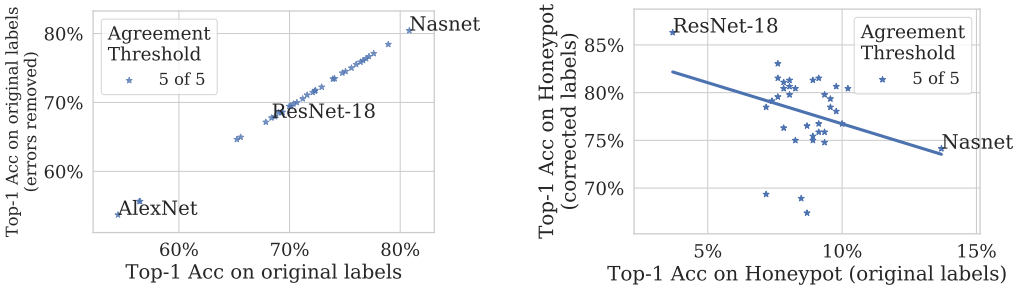
[1] https://github.com/cgnorthcutt/cleanlab

Each CL-identified label error was independently presented to five workers. We consider the example validated (an "error") if fewer than three of the workers agree that the data point has the given label (*agreement threshold = 3 of 5*) , otherwise we consider it to be a "non-error" (i.e. the original label was correct). We further categorize label errors, breaking them down into (1) "correctable", where a majority agree on the CL-predicted label; (2) "multi-label", where a majority agree on both labels appearing; (3) "neither", where a majority agree on neither label appearing; and (4) "non-agreement", a catch-all category for when there is no majority of workers. Beyond just validating whether potential errors are indeed errors, we categorize errors in this way because only the *correctable* errors may be used in a honeypot. Table 2 summarizes the results, and Figure 1 shows example label errors.

Table 2: Mechanical Turk validation confirming the existence of pervasive label errors.

Dataset	**Test Set Errors Categorization**					
	non-errors	errors	non-agreement	correctable	multi-label	neither
MNIST	85	15	2	10	-	3
CIFAR-10	221	54	32	18	0	4
CIFAR-100	1650	585	210	318	20	37
Caltech-256	335	65	25	22	5	13
ImageNet	2524	2916	598	1428	597	293
QuickDraw	630	1870	563	1047	20	240
20news	11	82	43	22	12	5
IMDB	585	725	552	173	-	-
Amazon	268	732	460	272	-	-
AudioSet	32	275	-	-	-	-

6 Case study: ImageNet, honeypots, and benefits of label errors



(a) ImageNet val set, with errors removed.

(b) ImageNet Honeypot errors, with corrected labels.

Figure 2: Benchmark ranking comparison of 34 models pre-trained on ImageNet val set (see Figure 4b for all agreement threshold settings and Table 3 for individual scores, in the Appendix). Benchmarks are unchanged by removing label errors (a), but change drastically on the Honeypot subset with original (erroneous) labels versus corrected labels, e.g. Nasnet re-ranking: 1/34 → 29/34, ResNet-18 re-ranking: 34/34 → 1/34. Test set sizes: original: 50,000; cleaned: 45,618; corrected: 460.

What are the consequences of test set label errors? Figure 2 compares performance on the ImageNet validation set, *commonly used in place of the test set*, of 34 pre-trained models from the PyTorch and Keras Python repositories. Figure 2a confirms the observations in [27]; benchmarks are unchanged by using a clean test set, i.e. in our case via the removal of errors.

However, we find a surprising result upon closer examination of models' performance *on the erroneously labeled data*, which we call the "honeypot set". When evaluating models *only* on the erroneously labeled data, models which perform best on the original (incorrect) labels perform the worst when those labels are corrected. This phenomenon, shown in Figure 2b, may indicate that many standard benchmark models trained on ImageNet had access to the validation set during training, perhaps in the form of hyper-parameter optimization, allowing these models to overfit to the validation set. Given that this set is commonly used in place of the test set, this is problematic.

We see this as a potential benefit of test set label errors: due to the presence of label errors in many benchmark datasets, this honeypot technique to identify overfitting to test sets is readily applicable to many datasets and trained models.

References

[1] Dana Angluin and Philip Laird. Learning from noisy examples. *Machine Learning*, 2(4):343–370, 1988.

[2] Pengfei Chen, Ben Ben Liao, Guangyong Chen, and Shengyu Zhang. Understanding and utilizing deep neural networks trained with noisy labels. In *International Conference on Machine Learning (ICML)*, 2019.

[3] Wikipedia contributors. List of datasets for machine learning research — wikipedia, the free encyclopedia, 2020. Online; accessed 22-October-2018.

[4] Alexander Philip Dawid and Allan M Skene. Maximum likelihood estimation of observer error-rates using the em algorithm. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 28(1):20–28, 1979.

[5] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[6] Jort F. Gemmeke, Daniel P. W. Ellis, Dylan Freedman, Aren Jansen, Wade Lawrence, R. Channing Moore, Manoj Plakal, and Marvin Ritter. Audio set: An ontology and human-labeled dataset for audio events. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, New Orleans, LA, 2017.

[7] Jacob Goldberger and Ehud Ben-Reuven. Training deep neural-networks using a noise adaptation layer. In *International Conference on Learning Representations (ICLR)*, 2017.

[8] G. Griffin, A. Holub, and P. Perona. Caltech-256 object category dataset. Technical Report 7694, California Institute of Technology, 2007.

[9] Patrick J Grother. Nist special database 19 handprinted forms and characters database. *National Institute of Standards and Technology*, 1995.

[10] David Ha and Douglas Eck. A neural representation of sketch drawings. *arXiv preprint arXiv:1704.03477*, 2017.

[11] Dan Hendrycks, Mantas Mazeika, Duncan Wilson, and Kevin Gimpel. Using trusted data to train deep networks on labels corrupted by severe noise. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2018.

[12] W Ronny Huang, Zeyad Emam, Micah Goldblum, Liam Fowl, Justin K Terry, Furong Huang, and Tom Goldstein. Understanding generalization through visualizations. *arXiv preprint arXiv:1906.03291*, 2019.

[13] Ishan Jindal, Matthew Nokleby, and Xuewen Chen. Learning deep networks from noisy labels with dropout regularization. In *International Conference on Data Mining (ICDM)*, 2016.

[14] Jan Kremer, Fei Sha, and Christian Igel. Robust active label correction. In Amos Storkey and Fernando Perez-Cruz, editors, *Proceedings of Machine Learning Research (PMLR)*, volume 84 of *Proceedings of Machine Learning Research*, pages 308–316, Playa Blanca, Lanzarote, Canary Islands, 09–11 Apr 2018. PMLR.

[15] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. The cifar-10 dataset. *online: http://www. cs. toronto. edu/kriz/cifar. htm*, 55, 2014.

[16] Yann Lecun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, pages 2278–2324, 1998.

[17] Zachary Lipton, Yu-Xiang Wang, and Alexander Smola. Detecting and correcting for label shift with black box predictors. In *International Conference on Machine Learning (ICML)*, 2018.

[18] Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. Learning word vectors for sentiment analysis. In *Annual Conference of the Association for Computational Linguistics (ACL)*, pages 142–150, Portland, Oregon, USA, June 2011. Annual Conference of the Association for Computational Linguistics (ACL).

[19] D. Mahajan, R. Girshick, V. Ramanathan, K. He, M. Paluri, Y. Li, A. Bharambe, and L. van der Maaten. Exploring the Limits of Weakly Supervised Pretraining. *ArXiv*, May 2018.

[20] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton van den Hengel. Image-based recommendations on styles and substitutes. In *Special Interest Group on Information Retrieval (SIGIR)*, pages 43–52, New York, NY, USA, 2015. ACM.

[21] Tom Mitchell. Twenty newsgroups dataset, 1999.

[22] Nagarajan Natarajan, Inderjit S Dhillon, Pradeep K Ravikumar, and Ambuj Tewari. Learning with noisy labels. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2013.

[23] Curtis G. Northcutt, Lu Jiang, and Isaac L. Chuang. Confident learning: Estimating uncertainty in dataset labels. *arXiv preprint arXiv:1911.00068*, 2019.

[24] Giorgio Patrini, Frank Nielsen, Richard Nock, and Marcello Carioni. Loss factorization, weakly supervised learning and label noise robustness. In *International Conference on Machine Learning (ICML)*, pages 708–717, 2016.

[25] Giorgio Patrini, Alessandro Rozza, Aditya Krishna Menon, Richard Nock, and Lizhen Qu. Making deep neural networks robust to label noise: A loss correction approach. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[26] Alexander J Ratner, Christopher M De Sa, Sen Wu, Daniel Selsam, and Christopher Ré. Data programming: Creating large training sets, quickly. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2016.

[27] Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do imagenet classifiers generalize to imagenet? In *International Conference on Machine Learning (ICML)*, pages 5389–5400, 2019.

[28] David Rolnick, Andreas Veit, Serge Belongie, and Nir Shavit. Deep learning is robust to massive label noise. *arXiv:1705.10694*, 2017.

[29] Sainbayar Sukhbaatar, Joan Bruna, Manohar Paluri, Lubomir Bourdev, and Rob Fergus. Training convolutional networks with noisy labels. In *International Conference on Learning Representations (ICLR)*, 2015.

[30] Chen Sun, Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta. Revisiting unreasonable effectiveness of data in deep learning era, 2017.

[31] Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Andrew Ilyas, and Aleksander Madry. From imagenet to image classification: Contextualizing progress on benchmarks. *arXiv preprint arXiv:2005.11295*, 2020.

[32] Brendan Van Rooyen, Aditya Menon, and Robert C Williamson. Learning with symmetric label noise: The importance of being unhinged. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2015.

[33] Colin Wei, Jason D Lee, Qiang Liu, and Tengyu Ma. On the margin theory of feedforward neural networks. *arXiv:1810.05369*, 2018.

[34] Yilun Xu, Peng Cao, Yuqing Kong, and Yizhou Wang. L_dmi: A novel information-theoretic loss function for training deep nets robust to label noise. In *Conference on Neural Information Processing Systems (NeurIPS)*, pages 6225–6236. Curran Associates, Inc., 2019.

[35] Jing Zhang, Victor S Sheng, Tao Li, and Xindong Wu. Improving crowdsourced label quality using noise correction. *IEEE transactions on neural networks and learning systems*, 29(5):1675–1688, 2017.

A Summary of confident learning (CL) for finding label errors

We briefly summarize CL joint estimation here. An unnormalized representation of the joint, 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 $\boldsymbol{\theta}$, and 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.

B Dataset details

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

MNIST [16]. 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 in order to copy a particular set of digits [9]. Label errors may arise from handwriting ambiguities; a digit may be written in a way that makes it impossible to determine what the given label is.

ImageNet [5]. ImageNet is a database of images and classes to benchmark image classification models. Images are scraped automatically by querying words from WordNet "synonym sets" (synsets) on image search engines. The images are labeled with Amazon Mechanical Turk workers who are asked whether each image contains objects of a particular given synset. Workers are 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.

IMDB [18]. The IMDB Large Movie Review Dataset is a collection of movie reviews to benchmark binary sentiment classification. The labels are automatically determined by the user's review (a score ≤ 10 out of ten is considered negative; ≥ 7 is considered positive).

QuickDraw [10]. 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;" however, the test dataset is partially or fully manually-labeled.

Amazon Reviews [20] The Amazon Reviews dataset is a collection of textual reviews and 5-star ratings from Amazon customers used to benchmark sentiment analysis models. **Modifications**: We

use 5-core (9.9 GB) variant of the dataset. 2-star and 4-star reviews are removed due to ambiguity with 1-star, and 5-star reviews, respectively, which if left in the dataset, could inflate error counts.

20news [21] 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 automatically gathering during the overall data collection procedure.

CIFAR-10 / CIFAR-100 [15] The CIFAR-10 / CIFAR-100 datasets are a collection of small 32×32 images and labels from a set of 10 or 100 classes, respectively. The images were automatically collected by querying for the class label. Human labelers were instructed to select images that matched their class label (query term). 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 [8]. Caltech-256 is a database of images and classes. Images were automatically downloaded 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.

AudioSet [6]. 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 detect of faint/non-salient audio events."

C Mechanical Turk study details

Figure 3 shows the Mechanical Turk worker interface, here shown with a data point from the CIFAR-100 dataset.

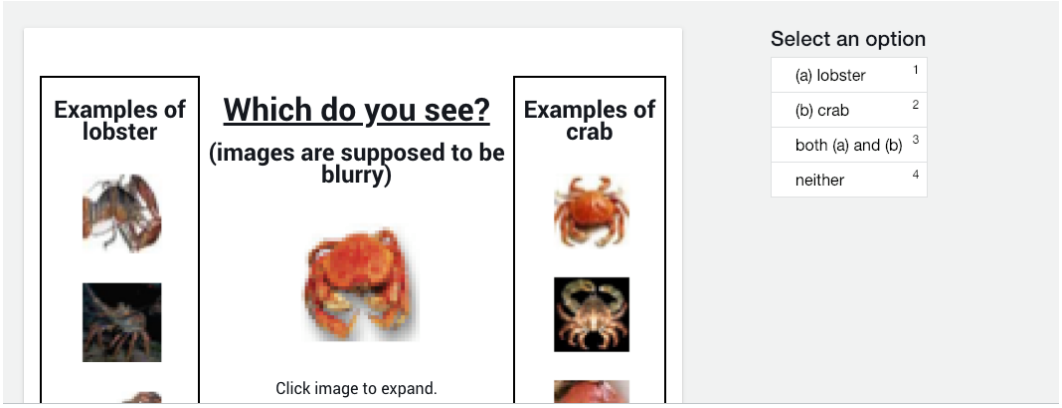
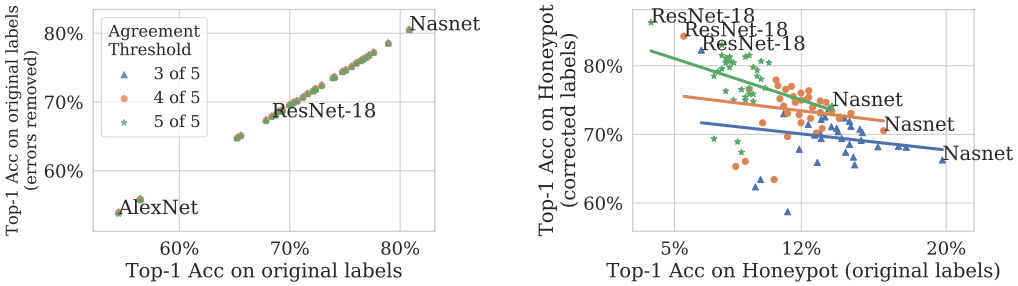


Figure 3: Mechanical Turk worker interface. For each image suspected of being a label error, the interface presents the image, along with example images belonging to the given class. The interface also shows images belonging to the confidently predicted class. Either the given is shown as (a) and predicted is shown as (b), or vice versa (chosen randomly). The worker is asked whether the image belongs to class (a), (b), both, or neither.

D Benchmarking details

Figure 2 depicts how the benchmarking rankings on the honeypot set of ImageNet examples change significantly for an *agreement threshold* $= 5$, meaning 5 of 5 human raters needed to independently select the same alternative label for that data point and 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 agreement threshold are shown in Figure 4b. Observe the negative correlation occurs in all three settings.



(a) ImageNet val set, with errors removed.

(b) ImageNet Honeypot errors, with corrected labels.

Figure 4: Benchmark ranking comparison of 34 models pre-trained on ImageNet val set for all settings of agreement threshold. Benchmarks are unchanged by removing label errors (a), but change drastically on the Honeypot subset with original (erroneous) labels versus corrected labels, e.g. Nasnet re-ranking: 1/34 $\rightarrow$ 29/34, ResNet-18 re-ranking: 34/34 $\rightarrow$ 1/34. Test set sizes: original (a): 50,000; cleaned (a): 47,114 (▲), 46,481 (●), 45,618 ($\star$); corrected (b): 1405 (▲), 943 (●), 460 ($\star$).

Table 3 provides the details of the accuracy scores and rankings for each model, for the setting of a majority agreement threshold (3 of 5).

Notably in Table 3, many rankings change drastically when benchmarked on the correct labels. For example, Nasnetlarge drops from ranking 1/34 to ranking 29/24, Xception drops from ranking 2/34 to 25/34, ResNet-18 increases from ranking 34/34 to ranking 1/34, and ResNet-50 increases from ranking 20/24 to 2/24. These dramatic changes in ranking may be explained by overfitting to the validation when these models are trained, which can occur inadvertently during hyper-parameter tuning. These results provide evidence that correct labels on a secret honey pot 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 3: Individual accuracy scores for Figure 4b with *agreement threshold = 3 of 5*. Acc@1 stands for the top-1 validation accuracy on the Honeypot set of original ImageNet examples and labels. *cAcc@1* stands for the top-1 val accuracy on the Honeypot 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 $\arg\max$ 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	6.48	82.28	73.88	99.57	34	1	28	1
PyTorch 1.0	resnet50	13.52	73.74	80.07	98.43	20	2	10	2
PyTorch 1.0	vgg19_bn	13.10	73.17	79.86	97.94	23	3	11	11
PyTorch 1.0	vgg11_bn	11.03	73.02	76.23	97.58	31	4	22	15
PyTorch 1.0	densenet169	14.16	72.53	79.79	98.29	16	6	12	3
PyTorch 1.0	resnet34	13.31	72.53	77.86	98.15	21	5	17	5
PyTorch 1.0	densenet121	14.38	72.38	78.58	97.94	14	7	16	9
PyTorch 1.0	vgg19	13.10	72.17	79.29	98.08	22	8	13	7
PyTorch 1.0	resnet101	14.66	71.89	81.28	98.22	12	9	5	4
PyTorch 1.0	vgg16	12.38	71.46	77.44	97.15	28	10	19	19
PyTorch 1.0	densenet201	14.73	71.17	80.78	97.94	10	11	6	10
PyTorch 1.0	vgg16_bn	13.67	71.10	77.72	97.51	19	12	18	16
Keras 2.2.4	densenet169	13.95	70.89	79.07	98.15	18	13	15	6
PyTorch 1.0	densenet161	15.23	70.75	80.28	98.01	7	14	8	8
Keras 2.2.4	densenet121	14.02	70.39	76.37	97.44	17	15	20	17
PyTorch 1.0	vgg11	13.02	70.25	75.30	97.22	25	17	27	18
PyTorch 1.0	resnet152	15.37	70.25	81.71	97.86	5	16	4	12
PyTorch 1.0	vgg13_bn	12.74	69.89	75.80	97.01	27	18	24	20
Keras 2.2.4	nasnetmobile	14.23	69.40	79.22	96.80	15	20	14	22
PyTorch 1.0	vgg13	13.10	69.40	76.23	96.80	24	19	21	23
Keras 2.2.4	densenet201	15.30	69.11	80.21	97.72	6	21	9	13
Keras 2.2.4	mobilenetV2	14.66	68.54	75.73	96.58	11	22	25	25
Keras 2.2.4	inceptionresnetv2	17.37	68.26	83.27	96.87	3	23	2	21
Keras 2.2.4	inceptionv3	16.23	68.19	80.36	96.73	4	24	7	24
Keras 2.2.4	xception	17.79	68.11	82.06	97.58	2	25	3	14
Keras 2.2.4	vgg19	11.89	67.83	73.81	95.52	29	26	30	30
Keras 2.2.4	mobilenet	14.45	67.40	73.74	96.09	13	27	31	27
Keras 2.2.4	resnet50	14.88	66.69	76.16	95.66	9	28	23	28
Keras 2.2.4	nasnetlarge	19.79	66.26	84.20	96.51	1	29	1	26
Keras 2.2.4	vgg16	12.88	65.91	73.81	95.66	26	30	29	29
PyTorch 1.0	inception_v3	14.95	65.55	75.59	95.30	8	31	26	31
PyTorch 1.0	squeezenet1_0	9.75	63.42	60.28	91.81	32	32	34	33
PyTorch 1.0	squeezenet1_1	9.47	62.35	61.64	92.31	33	33	33	32
PyTorch 1.0	alexnet	11.25	58.72	62.56	89.18	30	34	32	34