# Supplementary Materials of Towards Good Practices for Efficiently Annotating Large-Scale Image Classification Datasets

Yuan-Hong Liao[1,2], Amlan Kar[1,2,3], Sanja Fidler[1,2,3]

[1] University of Toronto, [2] Vector Institute, [3] NVIDIA

{andrew, amlan, fidler}@cs.toront.edu

## 1. More Ablations

### 1.1. Risk Threshold

The risk threshold $C$ in online annotation determines how fast the annotation process converges and possibly affects the final label quality. In Fig.1, we find that using a lower risk threshold stably results in stable label quality improvement. For larger risk threshold, some incorrectly annotated data can be considered confident in the early stage, resulting in unstable label quality improvement. It is somewhat surprising that varying the risk threshold does not affect the final label quality a lot. We set $C = 0.1$ for all other experiments in this paper.

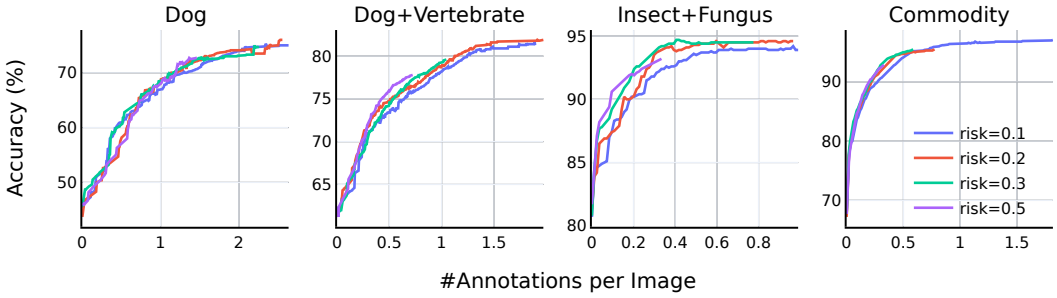



Figure 1: **Ablation of different risk thresholds.** Having lower risk threshold results in slower convergence, and improved final label quality.

### 1.2. Number of Workers

We explore how the number of workers affects annotation efficiency. In Fig.2, we find that having a fewer number of workers results in better label quality due to the better worker skills estimation, especially in the fine-grained dataset where the worker skills estimation matters more.

### 1.3. Model Update Frequency

In reality, we need to consider the frequency of updating the model and collecting human labels, and the latency of model updates varies across applications. In Fig.3, we find that having lower update frequencies (higher number of annotations per update) tends to overshoot in the number of annotations, while the method is robust to low update frequency.

### 1.4. Worker Prior Changes

There are multiple ways to design a better prior. Here, we discuss two possible ways: **A)** Considering class identity and **B)** Considering worker identity. To consider the class identity, the task designer needs to have a clear thought of which classes are harder. To consider the worker identity, the task designer needs to ask several gold standard questions to each worker. In

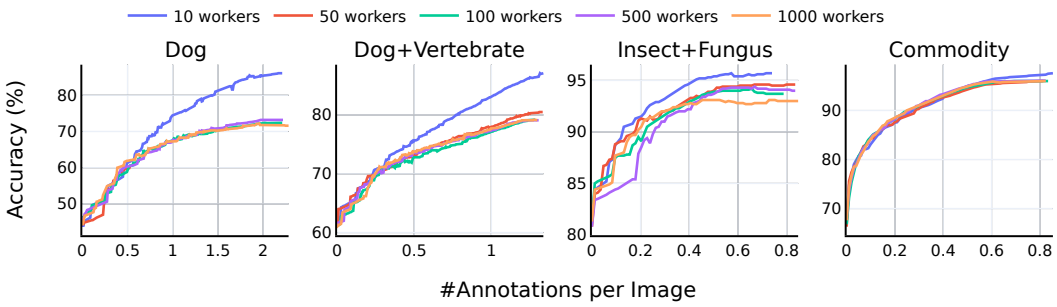



Figure 2: **Ablation of different numbers of workers.** Lower number of workers usually results in greater annotation quality but does not necessarily improve efficiency.

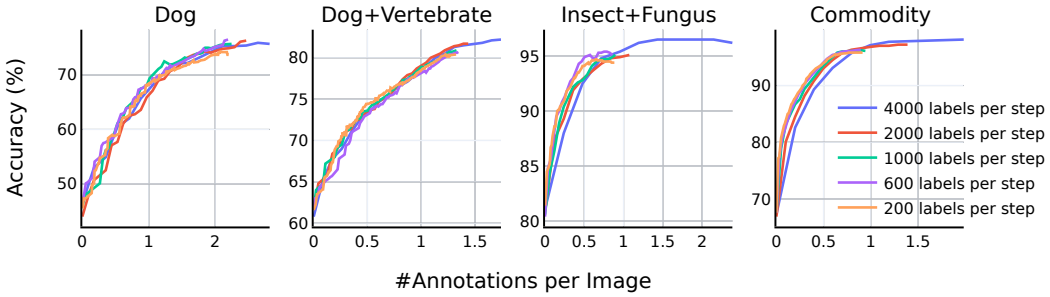



Figure 3: **Ablation of different numbers of labels per step.** Models that collect more labels per step tends to overhoot in number of annotations, while the final label quality remains similar.

Fig 4, we ablate the choice of having **None**, **A**, **B**, and **A+B** with prior strength 10. For the fine-grained datasets, considering worker identity in the prior improves a lot since the worker skills can vary significantly. For the coarse-grained datasets, the improvement is marginal. For all other experiments in this paper, we adopt **None**.

In reality, we need to consider the frequency of updating the model and collecting human labels, and the latency of model updates varies across applications. In Fig.3, we find that having lower update frequencies (higher number of annotations per update) tends to overshoot in the number of annotations, while the method is robust to low update frequency.

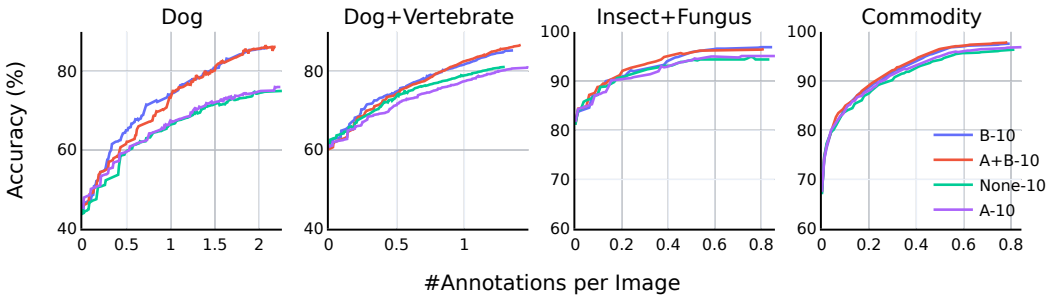



Figure 4: **Ablation of different priors.** For fine-grained dataset, considering worker identity in the prior improves a lot since the worker skills can vary a lot. However, having better prior does not necessarily lead to better performance.

### 1.5. Task Assignment

To verify if the learnt worker skills help with task assignment, we propose a simple greedy algorithm. For an image sampled for annotation from the unfinished set $\mathcal{U}$, we use $\bar{y_i}$ to find the best possible workers from the workers pool using the currently estimated worker skill, with a cap on the maximum number of annotations $\alpha$ allowed per worker.

$$j \leftarrow \arg\max_{j \in \hat{\mathcal{W}}} \mathbb{E}_{y \sim \bar{y_i}} \bar{w_j}[y, y], \hat{\mathcal{W}} = \{j || \mathcal{Z}_j | \leq \alpha\} \tag{1}$$

The worker importance is measured by a weighted sum of the worker's per-class reliability and the number of annotations assigned to the worker. The weights are inversely proportional to the model's per-class accuracy. Ideally, number worker annotations would be highly correlated with worker importance. In Fig. 5, we show the results on different splits .

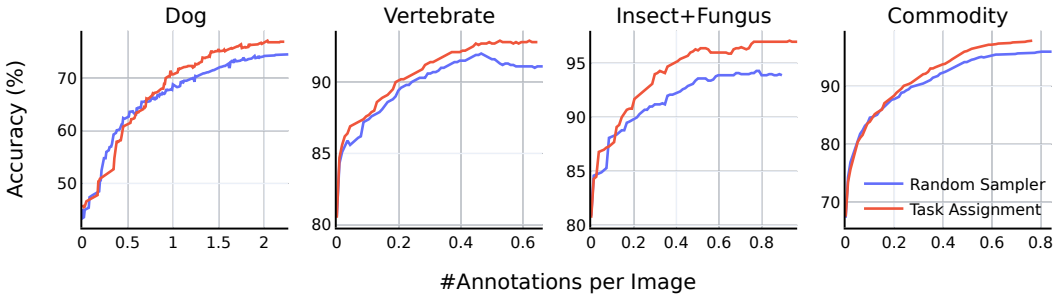



Figure 5: **Task Assignment.** The learnt skills consistently increase the annotation efficiency.

## 2. Transfer to Human Workers

To validate if the proposed approach can gracefully transfer to human workers, we further collect 2519 annotations of 1657 images from *Insesct+Fungus*. We perform data annotation from 11 workers on Toronto Annotation Suite. The average annotation accuracy is 0.908. For the experiments, we include 13000 additional images as unlabeled data for semi-supervised learning. Other details remain the same.

## 3. Data in the Unfinished Set

For all the experiments, we suggest performing early stopping and leave the rest of the unfinished set $\{x_i | \mathcal{R}(\bar{y_i} > C)\}$ to a separate process, possibly expert annotators. We show the normalized number of images in the unfinished set in Fig. 6. For coarse-grained datasets, there are almost no images left in the unfinished set, while in fine-grained datasets, *e.g. Dog*, there can be as many as 31% of images left in the unfinished set.

### 3.1. Precision of the Finished Set

In the main paper, we show the overall label accuracy. Here, we are interested in the precision of the finished set. In Fig 7, we measure the precision of the finished set of different methods and our proposed framework. In the *Dog*, the overall accuracy is 75.9%, while the precision of the finished set is 86.4%. The finished set size is 15598, *i.e.* we have 13368 correctly labeled images out of 22704 images in the finished set.

## 4. Top 5 in ImageNet100

The images in ImageNet can have multiple objects, making top1 accuracy unreliable. In Fig. 8, we show the comparison in terms of top5 accuracy. Our proposed approach still performs slightly better than its counterparts. Note that the top5 accuracy is more error-tolerant, making the improvement gap smaller.

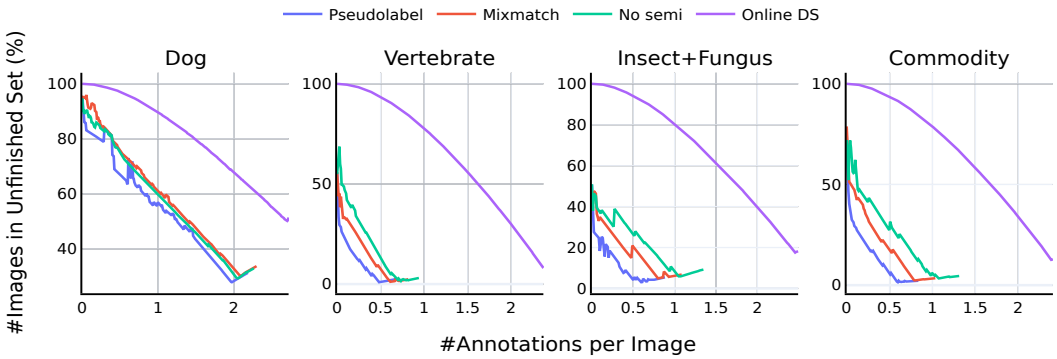



Figure 6: **Normalized number of images in the unfinished set.**

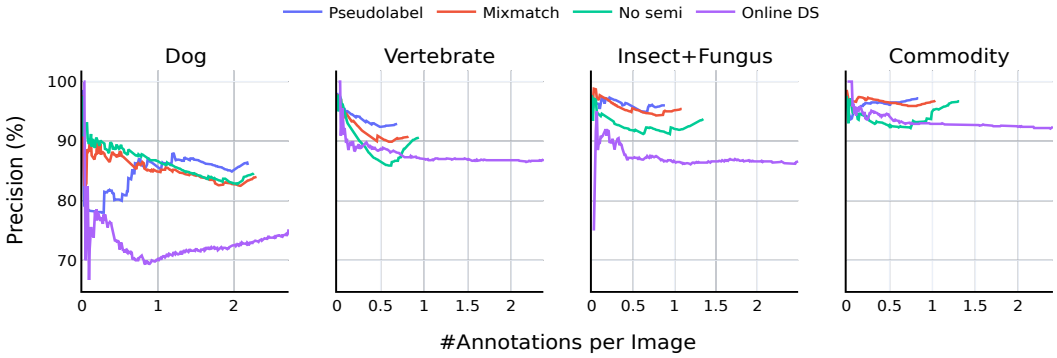



Figure 7: The precision of the finished set of images.

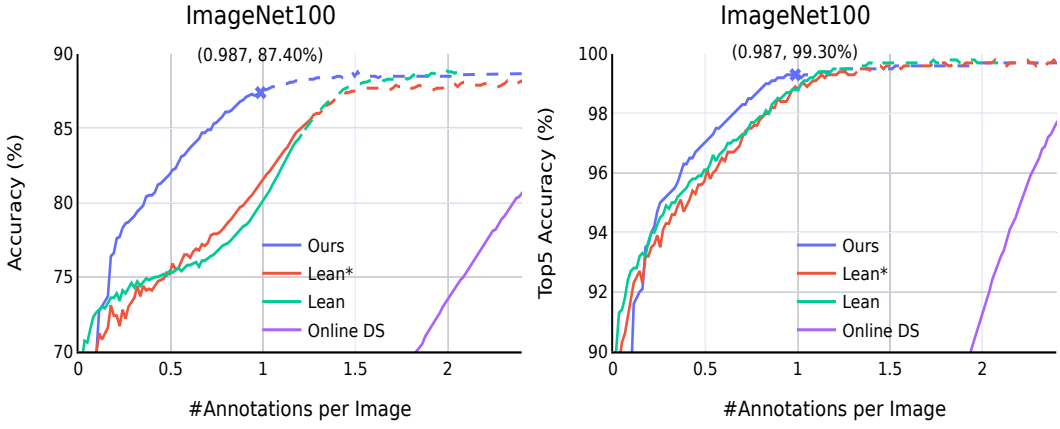



Figure 8: Top1 and top5 label accuracy on ImageNet100.

## 5. Implementation Details

For each experiment, the learning rate is $\lambda * \text{BatchSize}$, where $\lambda$ is the learning rate ratio. Regarding the worker prior, we follow previous work [1] to use a tiered prior system. Ignoring the worker identity, we use a homogeneous prior *i.e.* we ignore both worker identity and class identity. If gold standard questions are used, one can use a more sophisticated prior. We show the ablation of using different prior in Sec. 1.4. Each worker reliability is represented as a confusion matrix over

| Parameter | Search Values |
|---|---|
| $\lambda$ | $0.001, 0.0005, 0.0001, 0.00005$ |
| weight decay | $0.001, 0.005, 0.0005, 0.0001$ |
| $\mu$ | $3, 5, 10$ |
| $\gamma$ | $50, 75, 100, 150$ |

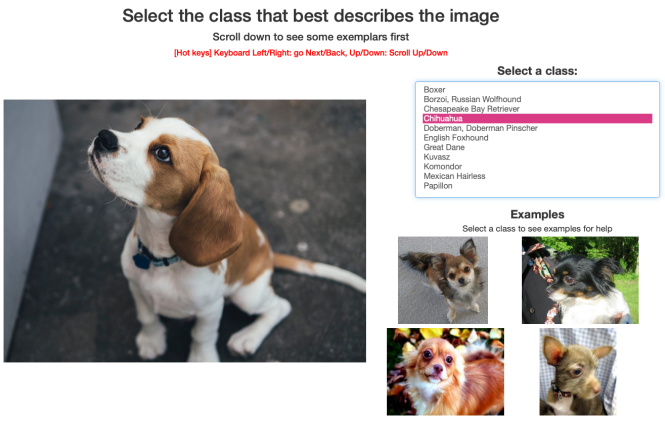
Table 1: The search range of the hyperparameters.

the label set. We assume a worker annotation $z$ given $y = k$ is a Dirichlet distribution $\text{Dir}(n_\beta \alpha_k)$, where $n_\beta$ is the strength of the prior and $\alpha_k$ is estimated by pooling across all workers and classes. We set $\alpha_k = 0.7$ and $n_\beta = 10$ for all experiments.

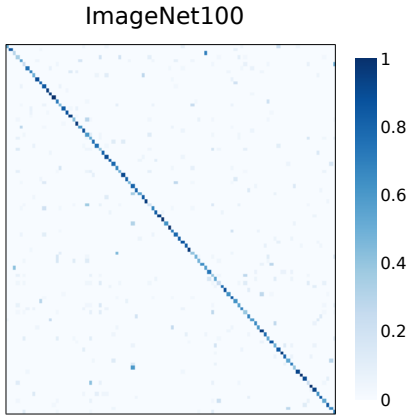
We perform hyperparameters search on learning rate ratio, weight decay ratio. For mixmatch, we perform an additional search over $\mu$, and $\gamma$. In Tab. 1, we show the search range of these hyperparameters.

## 6. Crowdsourcing on AMT

Prior work [2, 3] simulate workers as confusion matrices, and the class confusions are modeled with symmetric uniform noise, which can result in over-optimistic performance estimates. Human annotators usually exhibit *asymmetric* and *structured* confusion. We thus crowdsource the confusion matrices from human workers for the simulation. Fig. 9a, shows our user interface on Amazon Mechanical Turk (AMT) for crowdsourcing.




(a) The user interface on AMT for crowdsourcing.




(b) Average confusion matrix of all workers on the ImageNet-100 dataset

### 6.1. Workers Exhibit Structured Noise

We show the crowdsourced confusion matrices for different splits in Fig. 10. For coarse-grained datasets, *e.g*. *Commodity*, there is low confusion, while in fine-grained datasets (Rest), the confusions are strong and correlated to class identities. In Fig. 9b, we also show the average confusion matrix of all the workers on ImageNet100.

## 7. Simulating Workers

Here, we show the Python implementation to sample the simulated workers used in our experiments in Listing 1.

```
import numpy as np

def sample_confusion_matrix(smooth_ratio: float, noise_level: float, target_classes: list, imagenet100:
    list, group: list, group_workers: dict):

  cm = []
  for k, v in group_workers.items():
      v = np.array(v)
      global_v = v.sum(0)
```

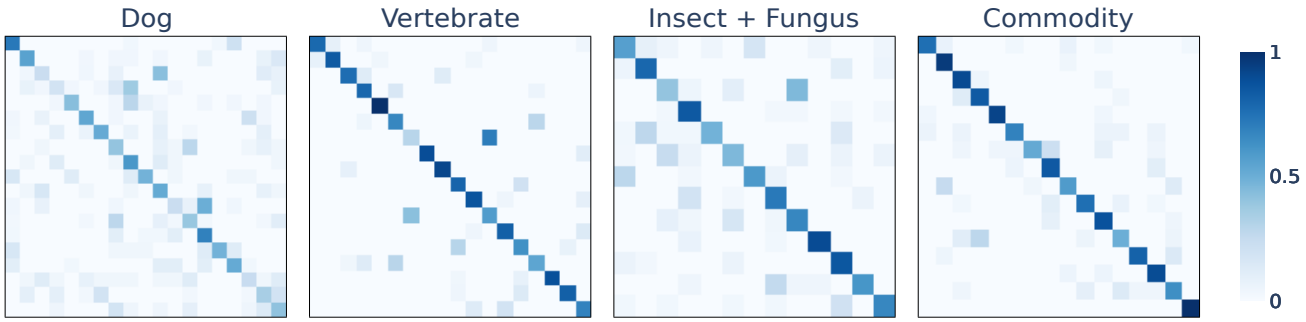



Figure 10: **The confusion matrix ImageNet100 splits.** We can see that the label confusion is neither unifom nor symmetric.

```
    idx = self.npr.choice(range(len(v)), 1)
    cm.append(smooth_ratio * global_v + v[idx])
cm = sum(cm)  # (100, 100) np.ndarray

idx_to_keep = np.array([imagenet100.index(c.lower()) for c in target_classes])
cm = cm[idx_to_keep, :][:, idx_to_keep]
cm = cm / (cm.sum(1, keepdims=True) + 1e-8)

def __which_group(i):
    for g_idx, g in enumerate(groups):
        if i in g:
            return g_idx


# Add uniform noise in off-diagonal terms
for i, c_i in enumerate(target_classes):
    c_i_group = __which_group(c_i)
    same_group_mask = np.zeros(cm.shape[0]).astype(np.bool)
    same_group_mask[i] = True
    for j, c_j in enumerate(target_classes):
        if i != j and c_i_group == __which_group(c_j):
            same_group_mask[j] = True

    if same_group_mask.sum() > 0:
        density_to_spread = cm[i, same_group_mask].sum()
        cm[i, same_group_mask] = cm[i, same_group_mask] * (1 - noise_level)
        cm[i, ~same_group_mask] += density_to_spread * (noise_level) / max(sum(~same_group_mask), 1e
  -8)

return cm
```

Listing 1: Worker Simulation Code

## 8. Semi-Supervised Learning: MixMatch

**MixMatch** constructs virtual training examples by mixing the labeled, and unlabeled data using a modified version of MixUp [6]. We modify MixMatch for online annotation, where the input to the model being learnt is the feature vector $\phi(x)$. The labeled set is defined by the data points with at least one worker annotation $\{x_i||\mathcal{W}_i| > 0\}$, and the unlabeled set is defined by the data points whose largest probability is larger than a predefined threshold and is not in the labeled set, $\{x_i|p(\bar{y}_i|\mathcal{Z}_i) > 1 - \tau \& |\mathcal{W}_i| = 0\}$. We use the same threshold as the one used in pseudo labels. The mixmatch loss consists of cross entropy of the labeled set and the l2 minimization of the mixed set. The mixed set is constructed by sampling $(x_1, p_1)$ from the labeled set and $(x_2, p_2)$ from the unlabeled set and interpolate both input and output.

$$
\begin{aligned}
\lambda &\sim \text{Beta}(\alpha, \alpha) \\
\lambda' &= \max(\lambda, 1-\lambda) \\
x' &= \lambda' x_1 + (1-\lambda') x_2 \\
p' &= \lambda' p_1 + (1-\lambda') p_2 \\
\mathcal{S}_{\text{mixed}} &\leftarrow (x', p') \\
L &= \mathbb{E}_{(x,y)\sim\{x_i,\bar{y_i}||\mathcal{W}_i|>0\}} H(\bar{y_i}, p(y|\phi(x_i),\theta)) \\
&\quad + \mu \mathbb{E}_{(x_i,p_i)\sim\mathcal{S}_{\text{mixed}}} \|p_i - p(y|\phi(x_i),\theta)\|_2^2
\end{aligned}
\tag{2}
$$

where $\mu$ is the hyperparameters. When the labeled set is small, we usually sample $\gamma$ times more data from the unlabeled set. In our experiments, $\mu$ and $\gamma$ are set by performing hyperparameters search mentioned in the main paper.

## 9. Unexplored Questions

We discuss shortcomings and additional directions for progress in this section.

**Using multiple ML models as workers:** Different self-supervised pretext tasks can provide orthogonal benefits for downstream tasks [4]. Downstream labeling tasks could go beyond semantic classes, such as annotating the viewing angle of a car in an image as a classification problem. One could imagine a scenario where classifiers trained on multiple self-supervised features are treated as machine "workers", whose skills for the task at hand are simultaneously estimated, similar to those for human workers.

**Annotating at a small scale / beyond ImageNet:** We do not discuss annotating small-scale data and restrict ourselves to ImageNet in this work. When the target dataset is small, we expect to be able to finetune self-supervised features on it to initialize the feature extractor $\phi$.

**Leveraging label hierarchies:** [5] propose a method to utilize label hierarchies to efficiently factorize large confusion matrices used to represent worker skills. We expect to see additional benefits from incorporating their skill estimation method into our algorithm.

**Simulating Image Difficulty:** Our proposed simulation does not account for image-level annotation difficulty, and simulated labels are obtained using a realistic confusion matrix applied to the ground truth label. Improving our simulation to consider this is something we would like to explore in future work.

**Going beyond classification:** The proposed method can be used to go beyond classification by changing the likelihood modeling for human annotations. In this vein, [1] show results on keypoint and bounding box annotation. Incorporating learning into the loop requires specific attention to detail per task, and we leave this to future work.

## 10. Class Names in Each Subtask

Tab. 2, shows the classes that comprise each subset in our ImageNet-sandbox dataset.

### 10.1. More Qualitative Results on ImageNet100

In Fig. 11, we show randomly sampled images from the unfinished set. Most of them are images with a fine-grained label, *e.g.* Fiddler Crab, Great Dane, Borzoi, etc, and some of them are shot from unusual angles, *e.g.* Chime and Basinet. We also show additional random examples with zero, one, two and three annotations in our ImageNet100 experiment in Fig. 12, Fig. 13, Fig. 14 and Fig. 15 respectively.

| Dataset | Class Names |
|---|---|
| Dog | Komondor, Mexican Hairless, Vizsla, Hungarian Pointer, Toy Terrier, Papillon, Boxer, Rottweiler, English Foxhound, Chihuahua, Shih-Tzu, Chesapeake Bay Retriever, Saluki, Gazelle Hound, Walker Hound, Walker Foxhound, Borzoi, Russian Wolfhound, Standard Poodle, Kuvasz, American Staffordshire Terrier, Staffordshire Terrier, American Pit Bull Terrier, Pit Bull Terrier, Doberman, Doberman Pinscher, Great Dane |
| Vertebrate | Meerkat, Mierkat, Hare, Robin, American Robin, Turdus Migratorius, Little Blue Heron, Egretta Caerulea, Tabby, Tabby Cat, Goose, Langur, Wild Boar, Boar, Sus Scrofa, Lorikeet, Garter Snake, Grass Snake, African Hunting Dog, Hyena Dog, Cape Hunting Dog, Lycaon Pictus, Gibbon, Hylobates Lar, Coyote, Prairie Wolf, Brush Wolf, Canis Latrans, Hognose Snake, Puff Adder, Sand Viper, American Coot, Marsh Hen, Mud Hen, Water Hen, Fulica Americana, Green Mamba, Gila Monster, Heloderma Suspectum, Red Fox, Vulpes Vulpes |
| Insect + Fungus | Gyromitra, Cauliflower, Fiddler Crab, Dung Beetle, Head Cabbage, American Lobster, Northern Lobster, Maine Lobster, Homarus Americanus, Stinkhorn, Carrion Fungus, Leafhopper, Rock Crab, Cancer Irroratus, Garden Spider, Aranea Diademata, Carbonara, Walking Stick, Walkingstick, Stick Insect, Chocolate Sauce, Chocolate Syrup |
| Commodity | Vacuum, Vacuum Cleaner, Computer Keyboard, Keypad, Bottlecap, Milk Can, Iron, Smoothing Iron, Mortarboard, Bonnet, Poke Bonnet, Sarong, Modem, Tub, Vat, Purse, Cocktail Shaker, Rotisserie, Jean, Blue Jean, Denim, Dutch Oven, Football Helmet |
| ImageNet20 | Robin, American Robin, Turdus Migratorius, Gila Monster, Heloderma Suspectum, Hognose Snake, Puff Adder, Sand Viper, Garter Snake, Grass Snake, Green Mamba, Garden Spider, Aranea Diademata, Lorikeet, Goose, Rock Crab, Cancer Irroratus, Fiddler Crab, American Lobster, Northern Lobster, Maine Lobster, Homarus Americanus, Little Blue Heron, Egretta Caerulea, American Coot, Marsh Hen, Mud Hen, Water Hen, Fulica Americana, Chihuahua, Shih-Tzu, Papillon, Toy Terrier, Walker Hound, Walker Foxhound, English Foxhound, Borzoi, Russian Wolfhound |

Table 2: Class names used in each split.


Figure 11: Images in the unfinished set on ImageNet100


Figure 12: Images with no annotations on ImageNet100

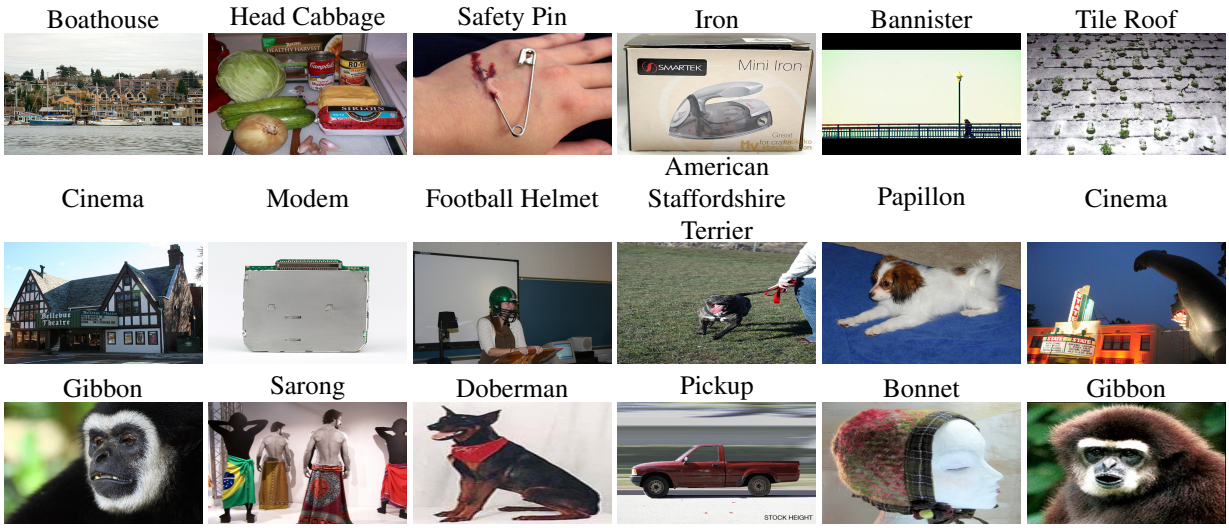



Figure 13: Images with 1 annotation on ImageNet100

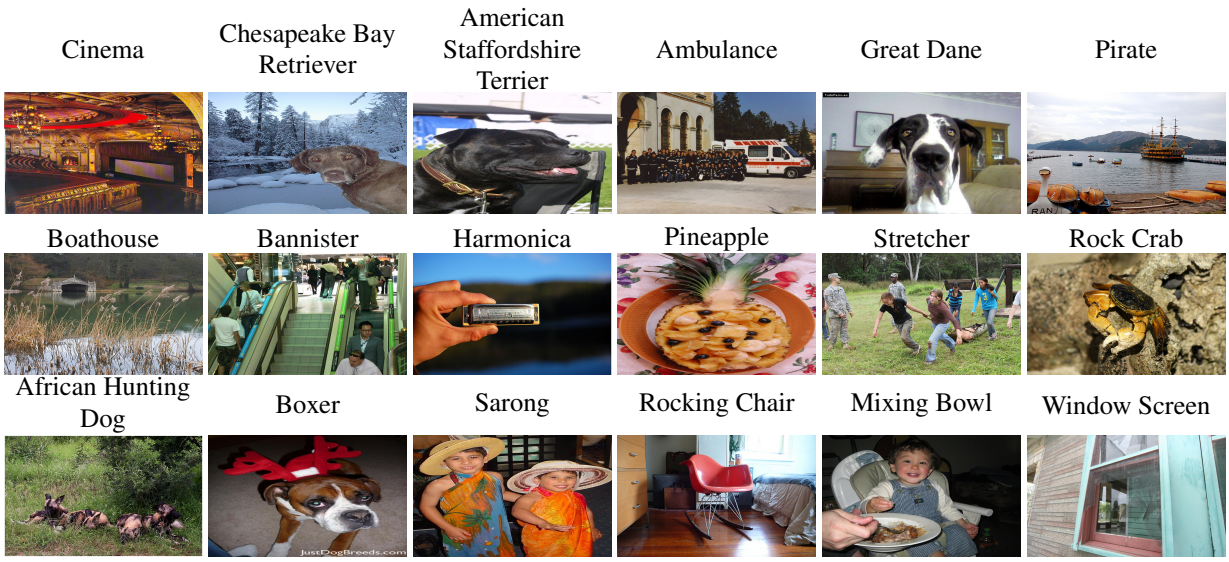



Figure 14: Images with 2 annotations on ImageNet100

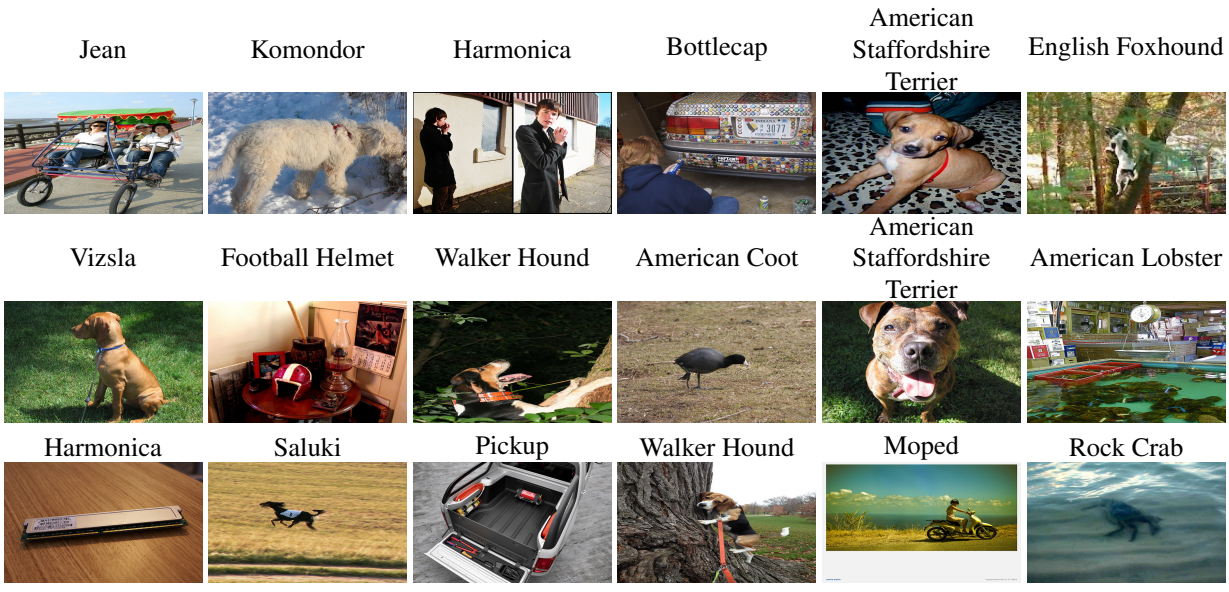



Figure 15: Images with 3 annotations on ImageNet100

## References


[1] Steve Branson, Grant Van Horn, and Pietro Perona. Lean crowdsourcing: Combining humans and machines in an online system. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[2] Gang Hua, Chengjiang Long, Ming Yang, and Yan Gao. Collaborative active learning of a kernel machine ensemble for recognition. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1209–1216, 2013.

[3] Chengjiang Long and Gang Hua. Multi-class multi-annotator active learning with robust gaussian process for visual recognition. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2839–2847, 2015.

[4] Yu Sun, Eric Tzeng, Trevor Darrell, and Alexei A Efros. Unsupervised domain adaptation through self-supervision. *arXiv preprint arXiv:1909.11825*, 2019.

[5] Grant Van Horn, Steve Branson, Scott Loarie, Serge Belongie, and Pietro Perona. Lean multiclass crowdsourcing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2714–2723, 2018.

[6] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.