Lean Crowdsourcing: Combining Humans and Machines in an Online System

Steve Branson
Caltech
sbranson@caltech.edu

Grant Van Horn
Caltech
gvanhorn@caltech.edu

Pietro Perona
Caltech
perona@caltech.edu

Abstract

We introduce a method to greatly reduce the amount of redundant annotations required when crowdsourcing annotations such as bounding boxes, parts, and class labels. For example, if two Mechanical Turkers happen to click on the same pixel location when annotating a part in a given image–an event that is very unlikely to occur by random chance–, it is a strong indication that the location is correct. A similar type of confidence can be obtained if a single Turker happened to agree with a computer vision estimate. We thus incrementally collect a variable number of worker annotations per image based on online estimates of confidence. This is done using a sequential estimation of risk over a probabilistic model that combines worker skill, image difficulty, and an incrementally trained computer vision model. We develop specialized models and algorithms for binary annotation, part keypoint annotation, and sets of bounding box annotations. We show that our method can reduce annotation time by a factor of 4-11 for binary filtering of websearch results, 2-4 for annotation of boxes of pedestrians in images, while in many cases also reducing annotation error. We will make an end-to-end version of our system publicly available.

1. Introduction

Availability of large labeled datasets like ImageNet [5, 21] is one of the main catalysts for recent dramatic performance improvement in computer vision [19, 12, 31, 32]. While sophisticated crowdsourcing algorithms have been developed for classification [46, 44, 45], there is a relative lack of methods and publicly available tools that use smarter crowdsourcing algorithms for other types of annotation.

We have developed a simple to use publicly available tool that incorporates and extends many recent advances in crowdsourcing methods to different types of annotation like part annotation and multi-object bounding box annotation, and also interfaces directly with Mechanical Turk.

Our main contributions are: 1) An online algorithm and stopping criterion for binary, part, and object crowdsourcing, 2) A worker skill and image difficulty crowdsourcing model for binary, part, and object annotations, 3) Incorporation of online learning of computer vision algorithms to speedup crowdsourcing, 4) A publicly available tool that interfaces with Mechanical Turk and incorporates these algorithms. We show that contributions 1–3 lead to significant improvements in annotation time and/or annotation quality for each type of annotation. For binary classification, annotation error with 1.37 workers per image is lower using our method than when using majority vote and 15 workers per image. For bounding boxes, our method produces lower error with 1.97 workers per image, compared to majority vote using 7 workers per image. For parts, a variation of our system without computer vision was used to annotate accurately a dataset of 11 semantic parts on 55,000 images, averaging 2.3 workers per part.

We note that while incorporating computer vision in the loop speeds up annotation time, computer vision re-

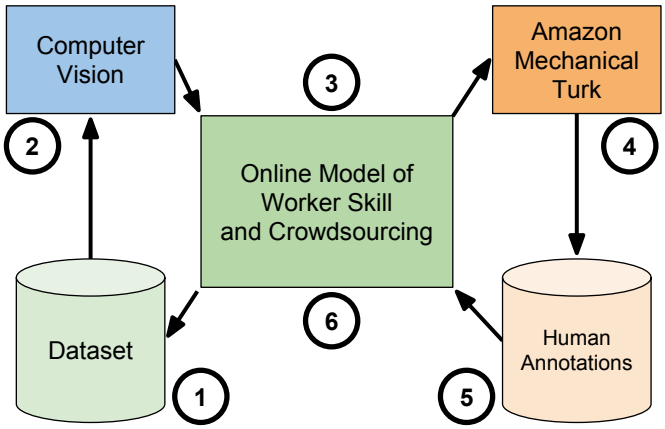


Figure 1: A schematic of our proposed method. **1)** The system is initialized with a dataset of images. Each global step of the method will add annotations to this dataset. **2)** The computer vision system incrementally retrains using current worker labels. **3)** The crowdsourcing model updates its predictions of worker skills and image labels and decides which images are finished based on a risk-based quality assurance threshold. Unfinished images are sent to Amazon Mechanical Turk. **4-5)** Workers on AMT annotate the images. **6)** The crowdsourcing model continues to update its predictions of worker skills and image labels, and the cycle is repeated until all images are marked as complete.

searchers wishing to collect datasets for benchmarking algorithms may choose to toggle off this option to avoid potential issues with bias. At the same time, we believe that it is a very valuable feature in applied settings. For example, a biologist may need to annotate the location of all cells in a dataset of images, not caring if the annotations come from humans or machines, but needing to ensure a certain level of annotation quality. Our method offers an end-to-end tool for collecting training data, training a prediction algorithm, combining human and machine predictions and vetting their quality, while attempting to minimize human time. This may be a useful tool for several applications.

2. Related Work

Kovashka et al. [17] provide a thorough overview of crowdsourcing in computer vision. Approaches that propose methods to combine multiple annotations with an assurance on quality are the most similar to our method. [29, 33] reconcile multiple annotators through majority voting and worker quality estimates. [45, 44, 23, 43] jointly model labels and the competence of the annotators. [13, 23, 22] explore the active learning regime of selecting the next data to annotate, as well as which annotator to query. Our approach differs from these previous methods by merging the online notion of [45] with the worker modeling of [44], and we incorporate a computer vision component as well as provide the framework for performing binary classification, bounding box and part annotations.

Our work is related to human-in-the-loop active learning. Prior work in this area has contributed methods for tasks such as fine-grained image classification [3, 40, 6, 42], image segmentation [26, 8, 10, 14], attribute-based classification [18, 24, 2], image clustering [20], image annotation [35, 36, 30, 48, 27], human interaction [16] and object annotation [39] and segmentation [28] in videos. For simplicity, we do not incorporate an active learning component when selecting the next batch of images to annotate or question to ask, but this can be included in our framework.

Additional methods to reduce annotation effort include better interfaces, better task organization [4, 7, 47], and gamifcation [37, 38, 15, 6].

3. Method

Let $X = \{x_i\}_{i=1}^N$ be a set of images we want to label with unknown true labels $Y = \{y_i\}_{i=1}^N$ using a pool of imperfect crowd workers. We first describe the problem generally, where depending on the desired application, each y_i may represent a class label, bounding box, part location, or some other type of semantic annotation. For each image i, our goal is to recover a label $\bar{y}_i$ that is equivalent to y_i with high probability by combining multiple redundant annotations $Z_i = \{z_{ij}\}_{j=1}^{|\mathcal{W}_i|}$, where each z_{ij} is an imperfect worker label (*i.e.*, their perception of y_i), and $\mathcal{W}_i$ is that set of workers that annotated image i.

Importantly, the number of annotations $|\mathcal{W}_i|$ can vary significantly for different images i. This occurs because our confidence on an estimated label $\bar{y}_i$ will depend not only on the number of redundant annotations $|\mathcal{W}_i|$, but also on the level of agreement between those annotations Z_i, the skill level of the particular workers that annotated i, and the agreement with a computer vision algorithm (that is incrementally trained).

3.1. Online Crowdsourcing

We first describe a simplified model that does not include a worker skill model or computer vision in the loop. We will augment this simplified model in subsequent sections. At any given time step, let $Z = \{Z_i\}_{i=1}^N$ be the set of worker annotations for all images. We define the probability over observed images, true labels, and worker labels as $p(Y, Z) = \prod_i p(y_i) \left(\prod_{j \in \mathcal{W}_i} p(z_{ij}|y_i) \right)$, where $p(y_i)$ is a prior probability over possible labels, and $p(z_{ij}|y_i)$ is a model of noisy worker annotations. Here we have assumed that each worker label is independent. The maximum likelihood solution $\bar{Y} = \arg\max p(Y|Z) = \arg\max p(Y, Z)$ can be found for each image separately: $\bar{y}_i = \arg\max_{y_i} \left(p(y_i) \prod_{j \in \mathcal{W}_i} p(z_{ij}|y_i) \right)$

The risk $\mathcal{R}(\bar{y}_i) = \int_{y_i} \ell(y_i, \bar{y}_i) p(y_i|Z_i)$ associated with the predicted label is

$$\mathcal{R}(\bar{y}_i) = \frac{\int_{y_i} \ell(y_i, \bar{y}_i) p(y_i) \prod_{j \in \mathcal{W}_i} p(z_{ij}|y_i)}{\int_{y_i} p(y_i) \prod_{j \in \mathcal{W}_i} p(z_{ij}|y_i)} \quad (1)$$

where $\ell(y_i, \bar{y}_i)$ is the loss associated with the predicted label $\bar{y}_i$ when the true label is y_i. A logical criterion is to accept $\bar{y}_i$ once the risk drops below a certain threshold $\mathcal{R}(\bar{y}_i) \leq \tau_\epsilon$ (*i.e.*, τ_ϵ is the minimum tolerable error per image). The basic online crowdsourcing algorithm, shown in Algorithm 1, processes images in batches (because sending images to services like Mechanical Turk is easier in batches). Currently, we give priority to annotating unfinished images with the fewest number of worker annotation $|\mathcal{W}_i|$; however, one could incorporate more sophisticated active learning criteria in future work. Each time a new batch is received, combined image labels $\bar{y}_i$ are re-estimated, and the risk criterion is used to determine whether or not an image is finished or may require more worker annotations.

3.2. Adding Computer Vision

A smarter algorithm can be obtained by using the actual pixel contents x_i of each image as an additional source of information. We consider two possible approaches: 1) a naive algorithm that treats computer vision the same way as a human worker by appending the computer vision prediction $z_{i,cv}$ to the set of worker labels $\mathcal{W}_i$, and 2) a smarter algorithm that exploits the fact that computer vision can provide additional information than a single label output

Algorithm 1 Online Crowdsourcing

input: unlabeled images $X = \{x_i\}_{i=1}^N$
Initialize unfinished/finished sets: $U \leftarrow \{i\}_{i=1}^N$, $F \leftarrow \emptyset$
Initialize $\bar{W}, \bar{I}$ using prior probabilities
repeat
  Select a batch $B \subseteq U$ of unfinished examples
  For $i \in B$ obtain new crowd label z_{ij}: $Z_i \leftarrow Z_i \cup z_{ij}$
  repeat ▷ Max likelihood estimation
    Estimate dataset-wide priors $p(d_i), p(w_j)$
    Predict true labels:
      $\forall_i,\ \bar{y}_i \leftarrow \arg\max_{y_i} p(y_i|x_i,\bar{\theta})p(Z_i|y_i,\bar{d}_i,\bar{W})$
    Predict image difficulties:
      $\forall_i,\ \bar{d}_i \leftarrow \arg\max_{d_i} p(d_i)p(Z_i|\bar{y}_i,d_i,\bar{W})$
    Predict worker parameters:
      $\forall_j,\ \bar{w}_j \leftarrow \arg\max_{w_j} p(w_j) \prod_{i\in\mathcal{I}_j} p(z_{ij}|\bar{y}_i,\bar{d}_i,w_j)$
  until Until convergence
  Using K-fold cross-validation, train computer vision on dataset $\{(x_i,\bar{y}_i)\}_{i,|\mathcal{W}_i|>0}$, and calibrate probabilities $p(y_i|x_i,\bar{\theta}_k)$
  Predict true labels:
    $\forall_i,\ \bar{y}_i \leftarrow \arg\max_{y_i} p(y_i|x_i,\bar{\theta})p(Z_i|y_i,\bar{d}_i,\bar{W})$
  for $i \in B$ **do** ▷ Check for finished labels
    $\mathcal{R}_i \leftarrow \frac{\int_{y_i} \ell(y_i,\bar{y}_i)p(\bar{y}_i|x_i,\bar{\theta})\prod_{j\in\mathcal{W}_i} p(z_{ij}|y_i,d_i,w_j)}{\int_{y_i} p(\bar{y}_i|x_i,\bar{\theta})\prod_{j\in\mathcal{W}_i} p(z_{ij}|y_i,d_i,w_j)}$
    if $\mathcal{R}_i \leq \tau_\epsilon$: $F \leftarrow F \cup i$, $U \leftarrow U \setminus i$
  end for
until $U = \emptyset$
return $Y \leftarrow \{\bar{y}_i\}_{i=1}^N$

(*e.g.*, confidence estimates that a bounding box occurs at each pixel location in an image).

For the smarter approach, the joint probability over observed images, true labels, and worker labels is:

$$p(Y,Z,\theta|X) = p(\theta)\prod_i \left(p(y_i|x_i,\theta) \prod_{j\in\mathcal{W}_i} p(z_{ij}|y_i) \right) \quad (2)$$

where $p(y_i|x_i,\theta)$ is the estimate of a computer vision algorithm with parameters θ.

Training Computer Vision: The main challenge is then training the computer vision system (estimating computer vision parameters θ) given that we incrementally obtain new worker labels over time. While many possible approaches could be used, in practice we re-train the computer vision algorithm each time we obtain a new batch of labels from Mechanical Turk. Each step, we treat the currently predicted labels $\bar{y}_i$ for each image with at least one worker label $|\mathcal{W}_i| \geq 1$ as training labels to an off-the-shelf computer vision algorithm. While the predicted labels $\bar{y}_i$ are clearly very noisy when the number of workers per image is still small, we rely on a post-training probability calibration step to cope with resulting noisy computer vision predictions. We use a modified version of K-fold cross validation: For each split k, we use $(K-1)/K$ examples for training and the remaining $(k-1)/K$ examples for probability calibration. We filter out images with $|\mathcal{W}_i| < 1$ from both training and probability calibration; however, all $1/K$ images are used for outputting probability estimates $p(y_i|x_i,\theta_k)$, including images with $|\mathcal{W}_i| = 0$. This procedure ensures that estimates $p(y_i|x_i,\theta_k)$ are produced using a model that wasn't trained on labels from image i.

3.3. Worker Skill and Image Difficulty Model

More sophisticated methods can model the fact that some workers are more skillful or careful than others and some images are more difficult or ambiguous than others. Let $W = \{w_j\}_{j=1}^M$ be parameters encoding the skill level of our pool of M crowd workers, and let $D = \{d_i\}_{i=1}^n$ be parameters encoding the level of inherent difficulty of annotating each image i (to this point, we are just defining W and D abstractly). Then the joint probability is

$$p(Y,Z,W,D,\theta|X) = p(\theta)\prod_i (p(d_i)p(y_i|x_i,\theta)) \prod_j p(w_j) \prod_{i,j\in\mathcal{W}_i} p(z_{ij}|y_i,d_i,w_j) \quad (3)$$

where $p(d_i)$ is a prior on the image difficulty, $p(w_j)$ is a prior on a worker's skill level, and $p(z_{ij}|y_i,d_i,w_j)$ models noisy worker responses as a function of the ground truth label, image difficulty and worker skill parameters. Let $\bar{Y},\bar{W},\bar{D},\bar{\theta} = \arg\max_{Y,W,D,\theta} p(Y,W,D,\theta|X,Z)$ be the maximum likelihood solution to Eq. 3: In practice, we estimate parameters using alternating maximization algorithms, where we optimize with respect to the parameters of one image or worker at a time (often with fast analytical solutions):

$$\bar{y}_i = \arg\max_{y_i} p(y_i|x_i,\bar{\theta}) \prod_{j\in\mathcal{W}_i} p(z_{ij}|y_i,d_i,w_j) \quad (4)$$

$$\bar{d}_i = \arg\max_{d_i} p(d_i) \prod_{j\in\mathcal{W}_i} p(z_{ij}|y_i,d_i,w_j) \quad (5)$$

$$\bar{w}_j = \arg\max_{w_j} p(w_j) \prod_{i\in\mathcal{I}_j} p(z_{ij}|\bar{y}_i,\bar{d}_i,w_j) \quad (6)$$

$$\bar{\theta} = \arg\max_{\theta} p(\theta) \prod_i p(\bar{y}_i|x_i,\theta) \quad (7)$$

where $\mathcal{I}_j$ is the set of images labeled by worker j. Exact computation of the risk $\mathcal{R}_i = \mathcal{R}(\bar{y}_i)$ is difficult because labels for different images are correlated through W and θ. An approximation is to assume our approximations $\bar{W}$, $\bar{I}$, and $\bar{\theta}$ are good enough $\mathcal{R}(\bar{y}_i) \approx \int_{y_i} \ell(y_i,\bar{y}_i)p(y_i|X,Z,\bar{\theta},\bar{W},\bar{D})$

$$\mathcal{R}(\bar{y}_i) \approx \frac{\int_{y_i} \ell(y_i,\bar{y}_i)p(y_i|x_i,\bar{\theta})\prod_{j\in\mathcal{W}_i} p(z_{ij}|y_i,\bar{d}_i,\bar{w}_j)}{\int_{y_i} p(y_i|x_i,\bar{\theta})\prod_{j\in\mathcal{W}_i} p(z_{ij}|y_i,\bar{d}_i,\bar{w}_j)}$$

such that Eq. 8 can be solved separately for each image i.

Considerations in designing priors: Incorporating priors is important to make the system more robust. Due to the online nature of the algorithm, in early batches the number of images $|\mathcal{I}_j|$ annotated by each worker j is likely small, making worker skill w_j difficult to estimate. Additionally, in practice many images will satisfy the minimum risk criterion with two or less labels $|\mathcal{W}_i| \leq 2$, making image difficulty d_i difficult to estimate. In practice we use a tiered prior system. A dataset-wide worker skill prior $p(w_j)$ and image difficulty prior $p(d_i)$ (treating all workers and images the same) is estimated and used to regularize per worker and per image parameters when the number of annotations is small. As a heuristic to avoid over-estimating skills, we restrict ourselves to considering images with at least 2 worker labels $|\mathcal{W}_i| > 1$ when learning worker skills, image difficulties and their priors, since agreement between worker labels is the only viable signal for estimating worker skill. We also employ a hand-coded prior that regularizes the learned dataset-wide priors.

4. Models For Common Types of Annotations

Algorithm 1 provides pseudo-code to implement the online crowdsourcing algorithm for any type of annotation. Supporting a new type of annotation involves defining how to represent true labels y_i and worker annotations z_{ij}, and implementing solvers for inferring the 1) true labels $\bar{y}_i$ (Eq. 4), 2) image difficulties $\bar{d}_i$ (Eq. 5), 3) worker skills $\bar{w}_j$ (Eq. 6), 4) computer vision parameters $\bar{\theta}$ (Eq. 7), and 5) risk $\mathcal{R}_i$ associated with the predicted true label (Eq. 8).

4.1. Binary Annotation

Here, each label $y_i \in 0, 1$, denotes the absence/presence of a class of interest.

Binary worker skill model: We model worker skill $w_j = [p_j^1, p_j^0]$ using two parameters representing the worker's skill at identifying true positives and true negatives, respectively. Here, we assume z_{ij} given y_i is Bernoulli, such that $p(z_{ij}|y_i = 1) = p_j^1$ and $p(z_{ij}|y_i = 0) = p_j^0$. As described in Sec 3.3, we use a tiered set of priors to make the system robust in corner case settings where there are few workers or images. Ignoring worker identity and assuming a worker label z given y is Bernoulli such that $p(z|y = 1) = p^1$ and $p(z|y = 0) = p^0$, we add Beta priors $\text{Beta}\left(n_\beta p^0, n_\beta(1-p^0)\right)$ and $\text{Beta}\left(n_\beta p^1, n_\beta(1-p^1)\right)$ on p_j^0 and p_j^1, respectively, where n_β is the strength of the prior. An intuition of this is that worker j's own labels z_{ij} softly start to dominate estimation of w_j once she has labeled more than n_β images, otherwise the dataset-wide priors dominate. We also place Beta priors $\text{Beta}\left(n_\beta p, n_\beta(1-p)\right)$ on p^0 and p^1 to handle cases such as the first couple batches of Algorithm 1. In our implementation, we use $p = .8$ as a general fairly conservative prior on binary variables and $n_\beta = 5$. This model results in simple estimation of worker skill priors $p(w_j)$ in line 8 of Algorithm 1 by counting the number of labels agreeing with combined predictions:

$$p^k = \frac{n_\beta p + \sum_{ij} 1[z_{ij} = \bar{y}_i = k, |\mathcal{W}_i| > 1]}{n_\beta + \sum_{ij} 1[\bar{y}_i = k, |\mathcal{W}_i| > 1]},\ k = 0, 1 \quad (8)$$

where $1[]$ is the indicator function. Analogously, we estimate worker skills w_j in line 11 of Algorithm 1 by counting worker j's labels that agree with combined predictions:

$$p_j^k = \frac{n_\beta p^k + \sum_{i \in \mathcal{I}_j} 1[z_{ij} = \bar{y}_i = k, |\mathcal{W}_i| > 1]}{n_\beta + \sum_{i \in \mathcal{I}_j} 1[\bar{y}_i = k, |\mathcal{W}_i| > 1]},\ k = 0, 1 \quad (9)$$

For simplicity, we decided to omit a notion of image difficulty in our binary model after experimentally finding that our simple model was competitive with more sophisticated models like CUBAM [44] on most datasets.

Binary computer vision model: We use a simple computer vision model based on training a linear SVM on features from a general purpose pre-trained CNN feature extractor (our implementation uses VGG), followed by probability calibration using Platt scaling [25] with the validation splits described in Sec. 3.2. This results in probability estimates $p(y_i|x_i, \theta) = \sigma(\gamma\, \theta \cdot \phi(x_i))$ for each image i, where $\phi(x_i)$ is a CNN feature vector, θ is a learned SVM weight vector, γ is probability calibration scalar from Platt scaling, and $\sigma()$ is the sigmoid function.

4.2. Part Keypoint Annotation

Part keypoint annotations are popular in computer vision and included in datasets such as MSCOCO [21], MPII human pose [1], and CUB-200-2011 [41]. Here, each part is typically represented as an x, y pixel location l and binary visibility variable v, such that $y_i = (l_i, v_i)$. While we can model v using the exact same model as for binary classification (Section 4.1), l is a continuous variable that necessitates different models. For simplicity, even though most datasets contain several semantic parts of an object, we model and collect each part independently. This simplifies notation and collection; in our experience, Mechanical Turkers tend to be faster/better at annotating a single part in many images than multiple parts in the same image.

Keypoint worker skill image difficulty model: Let l_i be the true location of a keypoint in image i, while l_{ij} is the location clicked by worker j. We assume l_{ij} is Gaussian distributed around l_i with variance σ_{ij}^2. This variance is governed by the worker's skill or image difficulty $\sigma_{ij}^2 = e_{ij}\sigma_j^2 + (1 - e_{ij})\sigma_i^2$, where σ_j^2 represents worker noise (*e.g.*, some workers are more precise than others) and σ_i^2 represents per image noise (*e.g.*, the precise location of a bird's belly in a given image maybe inherently ambiguous), and e_{ij} is a binary variable that determines if the variance will be governed by worker skill or image difficulty. However, worker j sometimes makes a gross mistake and clicks

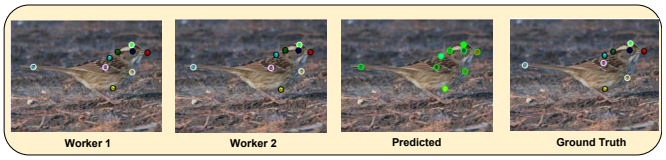


Figure 2: Example part annotation sequence showing the common situation where the responses from 2 workers correlate well and are enough for the system to mark the images as finished.

somewhere very far from the Gaussian center (*e.g.*, worker j could be a spammer or could have accidentally clicked an invalid location). m_{ij} indicates whether or not j made a mistake–with probability p_j^m–, in which case l_{ij} is uniformly distributed in the image. Thus

$$p(l_{ij}|y_i, d_i, w_j) = \sum_{m_{ij}\in 0,1} p(m_{ij}|p_j^m)p(l_{ij}|l_i, m_{ij}, \sigma_{ij}) \tag{10}$$

where $p(m_{ij}|p_j^m) = m_{ij}p_j^m + (1 - m_{ij})(1 - p_j^m)$, $p(l_{ij}|l_i, m_{ij}, \sigma_{ij}) = \frac{e_{ij}}{|x_i|} + (1 - e_{ij})g(\|l_{ij} - l_i\|^2; \sigma_{ij}^2)$, $|x_i|$ is the number of pixel locations in i, and $g(x^2; \sigma^2)$ is the probability density function for the normal distribution. In summary, we have 4 worker skill parameters $w_j = [\sigma_j, p_j^m, p_j^0, p_j^1]$ and one image difficulty parameter $d_i = \sigma_i$. As described in Sec 4.1, we place a dataset-wide Beta prior $\text{Beta}\left(n_\beta p^m, n_\beta(1 - p^m)\right)$ on p_j^m, where p^m is a worker agnostic probability of making a mistake and an additional Beta prior $\text{Beta}\left(n_\beta p, n_\beta(1 - p)\right)$ on p^m. Similarly, we place Scaled inverse chi-squared priors on σ_j^2 and σ_i^2, such that $\sigma_j^2 \sim \text{scale} - \text{inv}{-}\chi^2(n_\beta, \sigma^2)$ and $\sigma_i^2 \sim \text{scale} - \text{inv}{-}\chi^2(n_\beta, \sigma^2)$ where σ^2 is a dataset-wide variance in click location.

Inferring worker and image parameters: These priors would lead to simple analytical solutions toward inferring the maximum likelihood image difficulties (Eq. 5) and worker skills (Eq. 6), if m_{ij}, e_{ij}, and θ were known. In practice, we handle latent variables m_{ij} and e_{ij} using expectation maximization, with the maximization step over all worker and image parameters, such that worker skill parameters are estimated as

$$\sigma_i^2 = \frac{n_\beta\sigma^2 + \sum_{j\in\mathcal{W}_i}(1 - \mathbb{E}e_{ij})(1 - \mathbb{E}m_{ij})\|l_{ij} - l_i\|^2}{n_\beta + 2 + \sum_{j\in\mathcal{W}_i}(1 - \mathbb{E}e_{ij})(1 - \mathbb{E}m_{ij})} \tag{11}$$

$$\sigma_j^2 = \frac{n_\beta\sigma^2 + \sum_{i\in\mathcal{I}_j}\mathbb{E}e_{ij}(1 - \mathbb{E}m_{ij})\|l_{ij} - l_i\|^2}{n + 2 + \sum_{i\in\mathcal{I}_j}\mathbb{E}e_{ij}(1 - \mathbb{E}m_{ij})} \tag{12}$$

$$p_j^m = \frac{n_\beta p^m + \sum_{i\in\mathcal{I}_j}\mathbb{E}m_{ij}}{n_\beta + |\mathcal{I}_j|} \tag{13}$$

These expressions all have intuitive meaning of being like standard empirical estimates of variance or binomial parameters, except that each example might be soft-weighted by $\mathbb{E}m_{ij}$ or $\mathbb{E}e_{ij}$, and n_β synthetic examples have been added from the global prior distribution. Expectations are then

$$\mathbb{E}e_{ij} = \frac{g_j}{g_i + g_j}, \quad \mathbb{E}m_{ij} = \frac{1/|x_i|}{1/|x_i| + (1 - \mathbb{E}e_{ij})g_i + \mathbb{E}e_{ij}g_j}$$
$$g_i = g(\|l_{ij} - l_i\|^2; \sigma_i^2), \quad g_j = g(\|l_{ij} - l_i\|^2; \sigma_j^2) \tag{14}$$

We alternate between maximization and expectation steps, where we initialize with $\mathbb{E}m_{ij} = 0$ (*i.e.*, assuming an annotator didn't make a mistake) and $\mathbb{E}e_{ij} = .5$ (*i.e.*, assuming worker noise and image difficulty have equal contribution).

Inferring true labels: Inferring $\bar{y}_i$ (Eq. 4) must be done in a more brute-force way due to the presence of the computer vision term $p(y_i|x_i, \theta)$. Let $\mathbb{X}_i$ be a vector of length $|x_i|$ that stores a probabilistic part detection map; that is, it stores the value of $p(y_i|x_i, \theta)$ for each possible value of y_i. Let $\mathbb{Z}_{ij}$ be a corresponding vector of length $|x_i|$ that stores the value of $p(z_{ij}|y_i, d_i, w_j)$ at each pixel location (computed using Eq. 10[1]). Then the vector $\mathbb{Y}_i = \mathbb{X}_i \prod_{j\in\mathcal{W}_i}\mathbb{Z}_{ij}$ densely stores the likelihood of all possible values of y_i, where products are assumed to be computed using component-wise multiplication. The maximum likelihood label $\bar{y}_i$ is simply the argmax of $\mathbb{Y}_i$.

Computing risk: Let $\mathbb{L}_i$ be a vector of length $|x_i|$ that stores the loss $\ell(y_i, \bar{y}_i)$ for each possible value of y_i. We assume a part prediction is incorrect if its distance from ground truth is bigger than some radius (in practice, we compute the standard deviation of Mechanical Turker click responses on a per part basis and set the radius equal to 2 standard deviations). The risk associated with predicted label $\bar{y}_i$ according to Eq. 8 is then $\mathcal{R}_i = \mathbb{L}_i^T\mathbb{Y}_i/\|\mathbb{Y}_i\|_1$

4.3. Multi-Object Bounding Box Annotations

Similar types of models that were used for part keypoints can be applied to other types of continuous annotations like bounding boxes. However, a significant new challenge is introduced if multiple objects are present in the image, such that each worker may label a different number of bounding boxes, and may label objects in a different order. Checking for finished labels means ensuring not only that the boundaries of each box is accurate, but also that there are no false negatives or false positives.

Bounding box worker skill and image difficulty model: An image annotation $y_i = \{b_i^r\}_{r=1}^{|B_i|}$ is composed of a set of objects in the image where box b_i^r is composed of x,y,x2,y2 coordinates. Worker j's corresponding annotation $z_{ij} = \{b_{ij}^k\}_{k=1}^{|B_{ij}|}$ is composed of a potentially different number $|B_{ij}|$ of box locations with different ordering. However, if we can predict latent assignments $\{a_{ij}^k\}_{k=1}^{|B_{ij}|}$, where b_{ij}^k is worker j's perception of true box $b_i^{a_{ij}^k}$, we can model anno-

[1]In practice, we replace e_{ij} and m_{ij} with $\mathbb{E}e_{ij}$ and $\mathbb{E}m_{ij}$ in Eq. 10, which corresponds to marginalizing over latent variables e_{ij} and m_{ij} instead of using maximum likelihood estimates

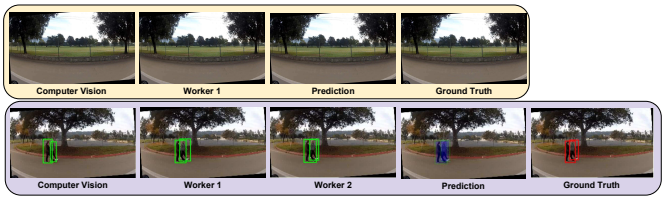


Figure 3: Bounding box annotation sequences. The top sequence highlights a good case where only the computer vision system and one human are needed to finish the image. The bottom sequence highlights the average case where two workers and the computer vision system are needed to finish the image.

tation of a matched bounding box exactly as for keypoints, where 2D vectors l have been replaced by 4D vectors b.

Thus as for keypoints the difficulty of image i is represented by a set of bounding box difficulties: $d_i = \{\sigma_i^r\}_{r=1}^{|B_i|}$, which measure to what extent the boundaries of each object in the image are inherently ambiguous. A worker's skill $w_j = \{p_j^{\text{fp}}, p_j^{\text{fn}}, \sigma_j\}$ encodes the probability p_j^{fp} that an annotated box b_{ij}^k is a false positive (*i.e.*, $a_{ij}^k = \emptyset$), the probability p_j^{fn} that a ground truth box b_i^r is a false negative (*i.e.*, $\forall_k, a_{ij}^k \neq r$), and the worker's variance σ_j^2 in annotating the exact boundary of a box is modeled as in Sec. 4.2. The number of true positives n_{tp}, false positives n_{fp}, and false negatives be n_{fn} can be written as $n_{\text{tp}} = \sum_{k=1}^{|B_{ij}|} 1[a_{ij}^k \neq \emptyset]$, $n_{\text{fn}} = |B_i| - n_{\text{tp}}$, $n_{\text{fp}} = |B_{ij}| - n_{\text{tp}}$. This leads to annotation probabilities

$$p(z_{ij}|y_i, d_i, w_j) = \prod_{k=1\ldots B_{ij}, a_{ij}^k \neq \emptyset} g\left(\left|b_i^{a_{ij}^k} - b_{ij}^k\right|^2; \sigma_{ij}^{k^2}\right) (p_j^{\text{fn}})^{n_{\text{fn}}}(1-p_j^{\text{fn}})^{n_{\text{tp}}}(p_j^{\text{fp}})^{n_{\text{fp}}}(1-p_j^{\text{fp}})^{n_{\text{tp}}} \quad (15)$$

As in the previous sections, we place dataset-wide priors on all worker and image parameters.

Computer vision: We train a computer vision detector based on MSC-MultiBox [32], which computes a shortlist of possible object detections and associated detection scores: $\{(b_{i,\text{cv}}^k, m_{i,\text{cv}}^k)\}_{k=1}^{|B_{i,\text{cv}}|}$. We choose to treat computer vision like a worker, with learned parameters $[p_{\text{cv}}^{\text{fp}}, p_{\text{cv}}^{\text{fn}}, \sigma_{\text{cv}}]$. The main difference is that we replace the false positive parameter $p_{\text{cv}}^{\text{fp}}$ with a per bounding box prediction of the probability of correctness as a function of its detection score $m_{i,\text{cv}}^k$. The shortlist of detections is first matched to boxes in the predicted label $\bar{y}_i = \{b_i^r\}_{r=1}^{|B_i|}$. Let $r_{i,\text{cv}}^k$ be 1 or -1 if detected box $b_{i,\text{cv}}^k$ was matched or unmatched to a box in $\bar{y}_i$. Detection scores are converted to probabilities using Platt scaling and the validation sets described in Sec. 3.2.

Inferring true labels and assignments: We devise an approximate algorithm to solve for the maximize likelihood label $\bar{y}_i$ (Eq. 4) concurrently with solving for the best assignment variables a_{ij}^k between worker and ground truth bounding boxes:

$$\bar{y}_i, a_i = \arg\max_{y_i, a_i} \log \sum_{j \in \mathcal{W}_i} \log p(z_{ij}|y_i, d_i, w_j) \quad (16)$$

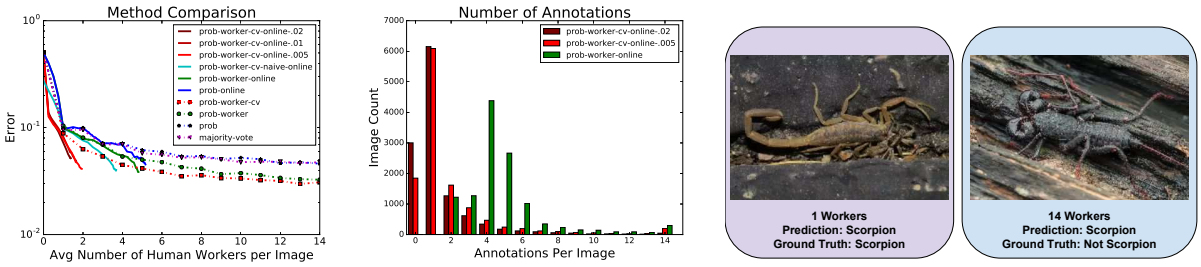
where $p(z_{ij}|y_i, d_i, w_j)$ is defined in Eq. 15. We formulate the problem as a facility location problem [9], a type of clustering problem where the objective is to choose a set of "facilities" to open up given that each "city" must be connected to a single facility. One can assign custom costs for opening each facility and connecting a given city to a given facility. Simple greedy algorithms are known to have good approximation guarantees for some facility location problems. In our formulation, facilities will be boxes selected to add to the predicted combined label $\bar{y}_i$ and city-facility costs will be costs associated with assigning a worker box to an opened box. Due to space limitations we omit derivation details; however, we set facility open costs $C^{\text{open}}(b_{ij}^k) = \sum_{j \in \mathcal{W}_i} - \log p_j^{\text{fn}}$ and city-facility costs $C^{match}(b_{ij}^k, b_{ij'}^{k'}) = -\log(1-p_j^{\text{fn}}) + \log p_j^{\text{fn}} - \log(1-p_j^{\text{fp}}) - \log g(\|b_{ij}^k - b_{ij'}^{k'}\|^2; \sigma_j^2)$ for matching worker box b_{ij}^k to facility $b_{ij'}^{k'}$, while not allowing connections where $j = j'$ unless $k = k', j = j'$. We add a dummy facility with open cost 0, such that cities matched to it correspond to worker boxes that are false positives: $C^{match}(b_{ij}^k, \text{dummy}) = -\log p_j^{\text{fp}}$.

Computing risk: We assume that the loss $\ell(\bar{y}_i, y_i)$ is defined as the number of false positive bounding boxes plus the number of false negatives, where boxes match if their area of intersection over union is at least 50%. To simplify calculation of risk (Eq. 8), we assume our inferred assignments a_{ij}^k between worker boxes and true boxes are valid. In this case, the risk $\mathcal{R}_i$ is the expected number of false positives (computed by summing over each b_i^r and computing the probability that it is a false positive according to Eq. 15), the expected number of true positives that were too inaccurate to meet the area of intersection over union criterion (computed using the method described in Section 4.2), and the expected number of false negatives in portions of the image that don't overlap with any true box b_i^r. Computation of the latter is included in supplementary details due to space limitations.

5. Experiments

We used a live version of our method to collect parts for the NABirds dataset. Additionally, we performed ablation studies on datasets for binary, part, and bounding box annotation based on simulating results from real-life MTurk worker annotations.

Evaluation Protocol: For each image, we collected an over-abundance of MTurk annotations per image which were used to simulate results by adding MTurk annotations in random order. For lesion studies, we crippled portions of Algorithm 1 as follows: 1) We removed online crowdsourcing by simply running lines 7-14 over the whole dataset



(a) Binary Method Comparison (b) Binary # Human Annotations (c) Binary Qualitative Examples

Figure 4: **Crowdsourcing Binary Classification Annotations: (a)** Comparison of methods. Our full model prob-worker-cvonline-0.02 obtains results as good as typical baselines with 15 workers (majority-vote and prob) using only 1.37 workers per image on average. **(b)** Histogram of the number of human annotations required for each image. **(c)** The image on the left represents an average annotation situation where only the computer vision label and one worker label are needed to confidently label the image. The image on the right (which is not a scorpion) represents a difficult case in which many workers disagreed on the label.

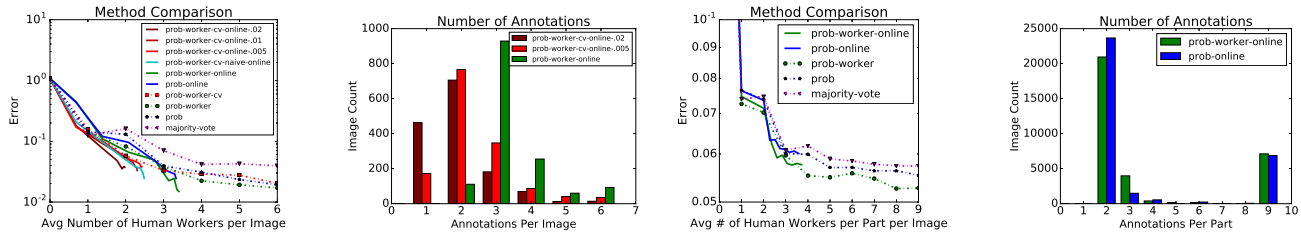
with k workers per image and sweeping over choices of k, 2) We removed the worker skill, image difficulty model by using dataset-wide priors, 3) We removed computer vision by using label priors $p(y_i)$ instead of computer vision estimates $p(y_i|x_i, \theta)$. As a baseline, the *majority-vote* method in plots 4a,5a,5c shows what we consider to be the most standard and commonly used method/baseline for crowdsourcing. For binary annotation, this selects the label with the most worker votes. For parts, it selects the median worker part location (*i.e.*, the one that matches the most other worker annotations with minimal loss). The same basic method is used for bounding boxes, adding a box if the majority of workers drew a box that could be matched to it. Fig. 4a,5a,5c show results for different lesioned methods. In each method name, the tag *worker* means that a worker skill and image difficulty model was used, the tag *online* means that online crowdsourcing was used (with parameter $\tau_\epsilon = .005$, unless a different number appears in the method name), the tag *cvnaive* means that a naive method to incorporate computer vision was used (by treating computer vision like a human worker, see Sec. 3.2), and the tag *cv* means that computer vision probabilities described in Sec. 4.1-4.2,4.3 were used.

Binary Annotation: We collected 3 datasets (scorpions, beakers, and cardigan sweaters) which we believe to be representative of the way datasets like ImageNet[5] and CUB-200-2011[41] were collected. For each category, we collected 4000 Flickr images by searching for the category name. 15 MTurkers per image were asked to filter search results. We obtained ground truth labels by carefully annotating images ourselves. Fig. 4a summarizes performance for the scorpion category (which is typical, see supplementary material for results on more categories), whereas Fig. 4c shows qualitative examples.

The full model prob-worker-cvonline-0.02 obtained results as good as typical baselines with 15 workers (majority-vote and prob) using only 1.37 workers per image on average. The method prob-online corresponds to the online crowdsourcing method of Welinder et al. [45], which used 5.1 workers per image and resulted in an error of 0.045; our full method prob-worker-cvonline-0.005 obtained lower error 0.041 with only 1.93 workers per image. We see that incorporating a worker skill model reduced converged error by about 33% (comparing prob-worker to majority-vote or prob). Adding online crowdsourcing roughly halved the number of annotations required to obtain comparable error (comparing prob-worker-online vs. prob-worker). Adding computer vision reduced the number of annotations per image by an additional factor of 2.4 with comparable error (comparing prob-worker-cvonline-0.005 to prob-worker-online). It also reduced annotations by a factor of 1.8 compared to the naive method of using computer vision (prob-worker-cvnaive-online), showing that using computer vision confidence estimates is useful. Interestingly, in Fig. 4b we see that adding computer vision allowed many images to be predicted confidently using no worker labels. Lastly, comparing prob-worker-cvonline-0.02 to prob-worker-cvonline-0.005, which resulted in errors of 0.051 and 0.041, respectively, and 1.37 vs. 1.93 workers per image, we see that the error tolerance parameter τ_ϵ offers an intuitive parameter to tradeoff annotation time and quality.

Bounding Box Annotation: To evaluate bounding box annotation, we used a 1448 image subset of the Caltech Roadside Pedestrian dataset [11]. We obtained ground truth annotations and 7 MTurk annotations per image from the creators of the dataset. We incur error for all false positives and negatives using a .5 IOU overlap criterion.

In Fig. 5a we see that the full model prob-worker-cvonline-0.02 obtained slightly lower error than majority-vote while using only 1.97 workers per image. This is encouraging given that most publicly available crowdsourcing



(a) BBox Method Comp (b) BBox # Human Annotations (c) Parts (d) Parts

Figure 5: **Crowdsourcing Multi-Object Bounding Box and Part Annotations:** **(a)** Our full model prob-worker-cvonline-0.02 obtains slightly lower error than majority-vote while using only 1.97 workers per image. **(b)** Histogram of the number of human annotators per image. **(c)** The worker skill model (prob-worker) led to 10% reduction in error over the majority-vote baseline, and the online model cut annotation time roughly in half. **(d)** Histogram of the number of human annotators per part.

tools for bounding box annotation use simple crowdsourcing methods. Incorporating a probabilistic model (comparing prob to majority-vote) reduced the error by a factor of 2, demonstrating that it is useful to account for probabilities of false positive and false negative boxes, and precision in drawing box boundaries. Online crowdsourcing reduced the number of required workers per image by a factor of 1.7 without increasing error (comparing prob-worker-online to prob-worker), while adding computer vision (method prob-worker-online-.005) reduced annotation by an additional 29%. Examining Fig. 5b, we see that computer vision allowed many images to be be confidently annotated with a single human worker. The naive computer vision method prob-worker-cvnaive-online performed as well as our more complicated method.

Part Annotation: To evaluate part keypoint annotation, we used the 1000 image subset of the NABirds dataset [34], for which a detailed analysis comparing experts to MTurkers was performed in [34]. This subset contained 10 MTurker labels per image of 11 semantic keypoint locations as well as expert part labels. Although our algorithm processed each part independently, we report error averaged over all 11 parts, using the loss defined in Sec. 4.2. We did not implement a computer vision algorithm for parts; however, a variant of our algorithm (prob-worker-online) was used by the creators of the dataset to collect its published part annotations (11 parts on 55,000 images), using only 2.3 worker annotations per part on average.

Simulated results on the 1000 image subset are shown in Fig. 5c. We see that the worker skill model (prob-worker) led to 10% reduction in error over the majority-vote baseline, and online model cut annotation time roughly in half, with most parts finishing with 2 worker clicks (Fig.4b)

Discussion and Failure Cases: All crowdsourcing methods resulted in some degree of error when crowd labels converged to something different than expert labels. The most common reason was ambiguous images. For example, most MTurkers incorrectly thought scorpion spiders (a type of spider resembling scorpions) were actual scorpions. Visibility of a part annotation can become ambiguous as an object rotates from frontal to rear view. However, all variants of our method (with and without computer vision, with and without online crowdsourcing) resulted in higher quality annotations than majority vote (which is commonly used for many computer vision datasets). Improvement in annotation quality came primarily from modeling worker skill. Online crowdsourcing can increase annotation errors; however, it does so with an interpretable parameter for trading off annotation time and error. Computer vision also reduces annotation time, with greater gains coming as dataset size increases.

6. Conclusion

In this work, we introduced crowdsourcing algorithms and online tools for collecting binary, part, and bounding box annotations. We showed that each component of the system–a worker skill / image difficulty model, an online stoppage criterion for collecting a variable number of annotations per image, and integration of computer vision in the loop–, each led to significant reductions in annotation time and/or annotation error for each type of annotation. In future work, we plan to extend the approach to other types of annotation like segmentation and video, use inferred worker skill parameters to block spammers or choose which worker should annotate an image, and incorporate active learning criteria to choose which images to annotate next or choose between different types of user interfaces.

Acknowledgments: This paper was inspired by work from and earlier collaborations with Peter Welinder and Boris Babenko. Much thanks to Pall Gunnarsson for helping to develop an early version of the method. Thank you to David Hall for supplying data for bounding box experiments. This work was supported by a Google Focused Research Award and Office of Naval Research MURI N000141010933.

References

[1] M. Andriluka, L. Pishchulin, P. Gehler, and B. Schiele. 4

[2] A. Biswas and D. Parikh. Simultaneous active learning of classifiers & attributes via relative feedback. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 644–651, 2013. 2

[3] S. Branson, C. Wah, F. Schroff, B. Babenko, P. Welinder, P. Perona, and S. Belongie. Visual recognition with humans in the loop. In *European Conference on Computer Vision*, pages 438–451. Springer, 2010. 2

[4] L. B. Chilton, G. Little, D. Edge, D. S. Weld, and J. A. Landay. Cascade: Crowdsourcing taxonomy creation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1999–2008. ACM, 2013. 2

[5] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. IEEE, 2009. 1, 7

[6] J. Deng, J. Krause, and L. Fei-Fei. Fine-grained crowdsourcing for fine-grained recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 580–587, 2013. 2

[7] J. Deng, O. Russakovsky, J. Krause, M. S. Bernstein, A. Berg, and L. Fei-Fei. Scalable multi-label annotation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 3099–3102. ACM, 2014. 2

[8] S. Dutt Jain and K. Grauman. Predicting sufficient annotation strength for interactive foreground segmentation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1313–1320, 2013. 2

[9] D. Erlenkotter. A dual-based procedure for uncapacitated facility location. *Operations Research*, 26(6):992–1009, 1978. 6

[10] D. Gurari, D. Theriault, M. Sameki, B. Isenberg, T. A. Pham, A. Purwada, P. Solski, M. Walker, C. Zhang, J. Y. Wong, et al. How to collect segmentations for biomedical images? a benchmark evaluating the performance of experts, crowdsourced non-experts, and algorithms. In *2015 IEEE Winter Conference on Applications of Computer Vision*, pages 1169–1176. IEEE, 2015. 2

[11] D. Hall and P. Perona. Fine-grained classification of pedestrians in video: Benchmark and state of the art. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5482–5491, 2015. 7

[12] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385*, 2015. 1

[13] G. Hua, C. Long, M. Yang, and Y. 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. 2

[14] S. D. Jain and K. Grauman. Active image segmentation propagation. CVPR, 2016. 2

[15] S. Kazemzadeh, V. Ordonez, M. Matten, and T. L. Berg. Referitgame: Referring to objects in photographs of natural scenes. In *EMNLP*, pages 787–798, 2014. 2

[16] M. Khodabandeh, A. Vahdat, G.-T. Zhou, H. Hajimirsadeghi, M. Javan Roshtkhari, G. Mori, and S. Se. Discovering human interactions in videos with limited data labeling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 9–18, 2015. 2

[17] A. Kovashka, O. Russakovsky, L. Fei-Fei, and K. Grauman. Crowdsourcing in Computer Vision. *ArXiv e-prints*, Nov. 2016. 2

[18] A. Kovashka, S. Vijayanarasimhan, and K. Grauman. Actively selecting annotations among objects and attributes. In *2011 International Conference on Computer Vision*, pages 1403–1410. IEEE, 2011. 2

[19] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012. 1

[20] S. Lad and D. Parikh. Interactively guiding semi-supervised clustering via attribute-based explanations. In *European Conference on Computer Vision*, pages 333–349. Springer, 2014. 2

[21] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *European Conference on Computer Vision*, pages 740–755. Springer, 2014. 1, 4

[22] C. Long and G. 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. 2

[23] C. Long, G. Hua, and A. Kapoor. Active visual recognition with expertise estimation in crowdsourcing. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3000–3007, 2013. 2

[24] A. Parkash and D. Parikh. Attributes for classifier feedback. In *European Conference on Computer Vision*, pages 354–368. Springer, 2012. 2

[25] J. Platt et al. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in large margin classifiers*, 10(3):61–74, 1999. 4

[26] M. Rubinstein, C. Liu, and W. T. Freeman. Annotation propagation in large image databases via dense image correspondence. In *European Conference on Computer Vision*, pages 85–99. Springer, 2012. 2

[27] O. Russakovsky, L.-J. Li, and L. Fei-Fei. Best of both worlds: human-machine collaboration for object annotation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2121–2131, 2015. 2

[28] N. Shankar Nagaraja, F. R. Schmidt, and T. Brox. Video segmentation with just a few strokes. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3235–3243, 2015. 2

[29] V. S. Sheng, F. Provost, and P. G. Ipeirotis. Get another label? improving data quality and data mining using multiple, noisy labelers. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 614–622. ACM, 2008. 2

[30] B. Siddiquie and A. Gupta. Beyond active noun tagging: Modeling contextual interactions for multi-class active learn-

ing. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 2979–2986. IEEE, 2010. 2

[31] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–9, 2015. 1

[32] C. Szegedy, S. Reed, D. Erhan, and D. Anguelov. Scalable, high-quality object detection. *arXiv preprint arXiv:1412.1441*, 2014. 1, 6

[33] T. Tian and J. Zhu. Max-margin majority voting for learning from crowds. In *Advances in Neural Information Processing Systems*, pages 1621–1629, 2015. 2

[34] G. Van Horn, S. Branson, R. Farrell, S. Haber, J. Barry, P. Ipeirotis, P. Perona, and S. Belongie. Building a bird recognition app and large scale dataset with citizen scientists: The fine print in fine-grained dataset collection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 595–604, 2015. 8

[35] S. Vijayanarasimhan and K. Grauman. Multi-level active prediction of useful image annotations for recognition. In *Advances in Neural Information Processing Systems*, pages 1705–1712, 2009. 2

[36] S. Vijayanarasimhan and K. Grauman. What's it going to cost you?: Predicting effort vs. informativeness for multi-label image annotations. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 2262–2269. IEEE, 2009. 2

[37] L. Von Ahn and L. Dabbish. Labeling images with a computer game. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 319–326. ACM, 2004. 2

[38] L. Von Ahn and L. Dabbish. Esp: Labeling images with a computer game. In *AAAI spring symposium: Knowledge collection from volunteer contributors*, volume 2, 2005. 2

[39] C. Vondrick, D. Patterson, and D. Ramanan. Efficiently scaling up crowdsourced video annotation. *International Journal of Computer Vision*, 101(1):184–204, 2013. 2

[40] C. Wah, S. Branson, P. Perona, and S. Belongie. Multiclass recognition and part localization with humans in the loop. In *2011 International Conference on Computer Vision*, pages 2524–2531. IEEE, 2011. 2

[41] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The caltech-ucsd birds-200-2011 dataset. 2011. 4, 7

[42] C. Wah, G. Van Horn, S. Branson, S. Maji, P. Perona, and S. Belongie. Similarity comparisons for interactive fine-grained categorization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 859–866, 2014. 2

[43] J. Wang, P. G. Ipeirotis, and F. Provost. Quality-based pricing for crowdsourced workers. 2013. 2

[44] P. Welinder, S. Branson, P. Perona, and S. J. Belongie. The multidimensional wisdom of crowds. In *Advances in neural information processing systems*, pages 2424–2432, 2010. 1, 2, 4

[45] P. Welinder and P. Perona. Online crowdsourcing: rating annotators and obtaining cost-effective labels. 2010. 1, 2, 7

[46] J. Whitehill, T.-f. Wu, J. Bergsma, J. R. Movellan, and P. L. Ruvolo. Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. In *Advances in neural information processing systems*, pages 2035–2043, 2009. 1

[47] M. J. Wilber, I. S. Kwak, and S. J. Belongie. Cost-effective hits for relative similarity comparisons. In *Second AAAI Conference on Human Computation and Crowdsourcing*, 2014. 2

[48] A. Yao, J. Gall, C. Leistner, and L. Van Gool. Interactive object detection. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 3242–3249. IEEE, 2012. 2