Best of both worlds: human-machine collaboration for object annotation
(preliminary version)

Olga Russakovsky
Stanford University
olga@cs.stanford.edu

Li-Jia Li
Snapchat Research
jia@snapchat.com

Li Fei-Fei
Stanford University
feifeili@cs.stanford.edu

Abstract

Despite the recent boom in large-scale object detection, the long-standing goal of localizing every object in the image remains elusive. The current best object detectors can accurately detect at most a few object instances per image. However, manually labeling objects is quite expensive.

This paper brings together the latest advancements in automatic object detection and in crowd engineering into a principled framework for accurately and efficiently localizing the objects in an image. Our proposed image annotation system seamlessly integrates multiple computer vision models with multiple sources of human input in a Markov Decision Process. The system is light-weight and easily extensible, performs self-assessment, and is able to incorporate novel objects instead of relying on a limited training set. It will be made publicly available.

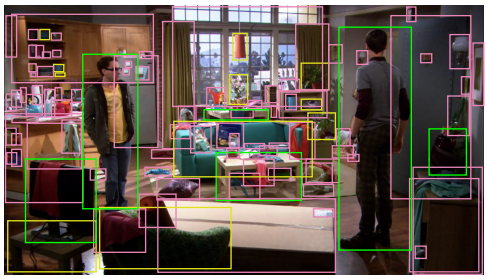

Figure 1. This cluttered image has 100 annotated objects shown with green, yellow and pink boxes. The green boxes correspond to the 6 objects correctly detected by the state-of-the-art RCNN model [20] trained on the ILSVRC dataset [41]. (The about 500 false positive detections are not shown.) Yellow boxes loosely correspond to objects that are annotated in current object detection datasets such as ILSVRC. The majority of the objects in the scene (shown in pink) are largely outside the scope of capabilities of current object detectors. We propose a principled human-in-the-loop framework for efficiently detecting all objects in an image.

1. Introduction

The field of large-scale object detection has leaped forward in the past few years [20, 41, 11, 43, 56, 23, 49], with significant progress both in techniques [20, 41, 49, 43] as well as scale: hundreds of thousands of object detectors can now be trained directly from web data [8, 15, 11]. The object detection models are commonly evaluated on benchmark datasets [41, 16], and achievements such as 1.9x improvement in accuracy between year 2013 and 2014 on the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [41] are very encouraging. However, taking a step back, we examine the performance of the state-of-the-art RCNN object detector trained on ILSVRC data [20] on the image of Figure 1: only the 6 green objects out of the 100 annotated objects have been correctly detected.

The question we set out to address in this paper: what can be done to efficiently and accurately detect all objects in an image given the current object detectors? One option is by utilizing the existing models for total scene understanding [34, 58, 33] or for modeling object context [59, 14, 42, 45]. However, this is still currently not enough to go from detecting 6 to detecting 100 objects.

Our answer is to put humans in the loop. The field of crowd engineering has provided lots of insight into human-machine collaboration for solving difficult problems in computing such as protein folding [39, 9], disaster relief distribution [18] and galaxy discovery [36]. In computer vision with human-in-the-loop approaches, human intervention has ranged from binary question-and-answer [5, 6, 55] to attribute-based feedback [38, 37, 32] to free-form object annotation [54]. For understanding all objects in an image, one important decision is which questions to pose to users. Binary questions are not sufficient. Asking users to draw bounding boxes is expensive: obtaining an accurate box around a single object takes between 7 seconds [25] to 42 seconds [47], and with 23 objects in an average indoor scene [21] the costs quickly add up. Based on insights from object detection dataset construction [35, 41], it is best to use a variety of human interventions; however, trading off accuracy and cost of labeling becomes a challenge.

We develop a principled framework integrating state-of-the-art scene understanding models [20, 31, 1, 21] with state-of-the-art crowd engineering techniques [41, 35, 10, 44, 27] for detecting objects in images. We formulate the optimization as a Markov Decision Process. Our system:

1. **Seamlessly integrates computer and human input**, accounting for the imperfections of both sources. [5, 25] One of the key components of our system, in contrast to prior work, is the ability to incorporate feedback from multiple types of human input and from multiple computer vision models (Section 4.1).

2. **Is open-world**, by integrating novel types of scenes and objects instead of relying only on information available in a limited training set (Section 4.2).

3. **Automatically trades off density, precision and cost of labeling.** Different scenarios might have unique requirements for the detected objects. Our system provides a principled way of detecting the important objects under the specified constraints (Section 4.4).

4. **Is light-weight and easily extensible**. Over time, as computer vision models become more accurate and novel crowdsourcing tools reduce the human time and cost, the framework will effortlessly incorporate the latest advances to continue providing the optimal labeling approach. *All code will be publicly available.*

We provide insights into seven types of human interventions tasks using data collected from Amazon Mechanical Turk (Section 5.2), and experimentally verify that our system effectively takes advantage of multiple sources of input for localizing objects in images (Section 5.3) while accurately self-monitoring (Section 5.4).

2. Related work

Recognition with humans in the loop. Among the most similar works to ours is the approaches which combine computer vision with human-in-the-loop collaboration for tasks such as fine-grained image classification [5, 6, 12, 55], image segmentation [25], attribute-based classification [30, 38, 3], image clustering [32], image annotation [50, 51] or object labeling in videos [54]. Methods such as [5, 6, 12, 55] jointly model human and computer uncertainty and characterize human labeling cost versus labeling accuracy, but only incorporate a single type of user response and terminate user labeling when budget is exceeded. Works such as [25, 13, 50] use multiple modalities of user feedback, with varying costs, and accurately model and predict the success of each modality. However, they do not incorporate iterative improvement in labeling. We build upon these approaches to integrate multiple modalities of crowdsourcing annotation directly with state-of-the-art computer vision models in an iterative framework, alternating between computer vision optimization and additional human labeling as needed, for the challenging task of dense object labeling.

Better object detection. Methods have been developed for training better object detection models with weakly supervised data [40, 22, 48, 8, 23, 15]. Active learning approaches has been developed to improve object detectors with minimal human annotation cost during training [30, 52]. Some object detection frameworks even automatically mine the web for object names and exemplars [8, 11, 15]. All of these approaches can be plugged into our framework to reduce the need for human annotation by substituting more accurate automatic detections.

Cheaper manual annotation. Manual annotation is becoming cheaper and more effective through the development of crowdsourcing techniques such as annotation games [53, 12, 29], tricks to reduce the annotation search space [13, 4], designing more effective user interfaces [47, 54], making use of weak user supervision [25, 7] and determining the minimum number of workers required for accurate labeling [44]. These innovations are important in our framework for minimizing the cost of human labeling when it is needed to augment computer vision. Approaches such as [10, 44, 27, 57] use iterative improvement to perform a task with highest utility, defined as accuracy of labeling per human cost. Methods such as [60] can be used to predict where human intervention is needed. We draw upon these works to provide human feedback in the most effective way.

3. Problem formulation

We present a policy for efficiently and accurately detecting objects in a given image. There are three inputs to the system. The first input is an **image to annotate**. The second input is a **utility function**, mapping each image region and class label to a real value indicating how much the system should care about this label: some objects in the image may be more important than others [46, 24, 2]. For example, if the goal is to label as many unique objects as possible then the utility would be the number of correct returned bounding boxes that correspond to unique object instances.

The third input is a set of **constraints on the annotation**. There are three possible constraints: (1) *utility*: having specified the utility function, the requester can set the desired utility value for the image labeling; (2) *precision*: after the system returns a set of N bounding box annotations with object names, if N_C of them are correct detections then precision is $\frac{N_C}{N}$; (3) *budget*: in our formulation, budget corresponds to cost of human time although methods such as [54] can be applied to also incorporate CPU cost. The requester may specify up to two constraints.

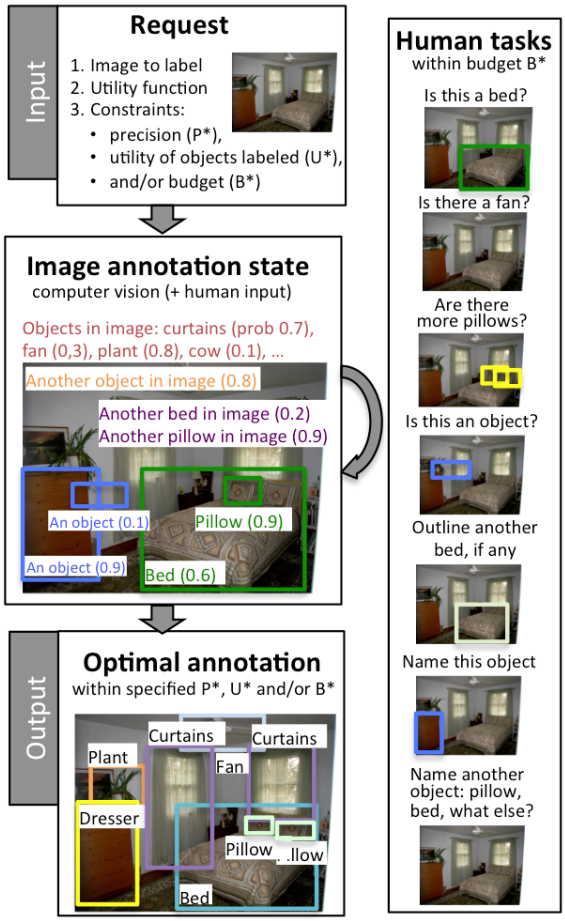

Figure 2. Overview of our system. Given a request for labeling an image, the system alternates between updating the image annotation and soliciting user feedback through human tasks. Upon satisfying the requester specifications, it terminates and returns a image with a set of bounding box annotations.

The goal of the system is to label the objects in the image under these constraints. The output is a set of bounding box annotations with object names. On one end of the spectrum the requester can set the maximum budget to zero, and obtain the best computer vision annotation of the image. On the other end she can set an infinite budget but specify 100% desired precision and 17 annotated objects per image, which will produce a policy for dense accurate labeling similar to that of the SUN dataset [59].

4. Method

Given a request for labeling objects in an image subject to some constraints, the system annotates the image using both computer vision and user input (Figure 2). It alternates between updating the image annotation state (Section 4.1) and getting feedback from human tasks (Section 4.2). The system terminates upon achieving a labeling that satisfies the request (Section 4.3). Until then it continues selecting the optimal human feedback (Section 4.4).

For the rest of this paper, we distinguish the requester (the person who wants the image annotated) from the users (the people doing the human labeling tasks).

4.1. Image annotation state

To effectively label objects in an image, our system uses information from both image appearance I and user feedback $\mathcal{U}$. The system maintains five types of estimates:

1. **Classification:** $P(\text{cls}(C)|I,\mathcal{U})$ for an object class C is the probability that this object is present in the image
2. **Detection:** $P(\text{det}(B,C)|I,\mathcal{U})$ for a bounding box B and object class C is the probability that box B is a good bounding box around an instance of class C
3. **Another instance:** $P(\text{moreinst}(\mathcal{B},C)|I,\mathcal{U})$ for a set of bounding boxes $\mathcal{B}$ and an object class C is the probability that there are other instances of class C in the image beyond those contained in $\mathcal{B}$
4. **Another class:** $P(\text{morecls}(\mathcal{C}|I,\mathcal{U}))$ for another object class in the image beyond the classes in set $\mathcal{C}$
5. **New object:** $P(\text{newobj}(B)|I,\mathcal{U})$ for a bounding box B is the probability that there is an object in this box which has not yet been named in the current annotation

The last three values are not commonly used in computer vision systems but are important when soliciting user feedback: for example, if there is likely to be another instance of an object class in the image, the user can be asked to draw one. Additional sources of information such as relative object layout can also be incorporated into the model.

We now describe how to use computer vision and then user information to estimate these probabilities.

4.1.1 Computer vision input

We incorporate four types of computer vision information into our system. The first two are straight-forward: **image-level classifiers**, providing $P(\text{cls}(C)|I)$ for every object class C and **object detectors**, providing $P(\text{det}(B,C)|I)$ for box B and class C. These models can be learned using a variety of computer vision methods [20, 17, 31, 23]. We use non-maximum suppression on the output of object detectors to (1) avoid multiple detections around the same instance, and (2) reduce the computational burden.

The third type of input is an empirical **distribution on number of object instances** in images. It provides the probability $P(\text{moreinst}|n)$ of there being more instances of an object class given the image is known to contain at least n instances of this class. If our system is considering the set $\mathcal{B}$ boxes for class C and we let $\text{nc}(\mathcal{B},C)$ be the

number of these boxes that are correct, then $\mathbb{E}[\text{nc}(\mathcal{B}, C)] = \sum_{B \in \mathcal{B}} P(\text{det}(B, C)|I)$. Rounding $n := \mathbb{E}[\text{nc}(\mathcal{B}, C)]$ to the nearest integer, we compute

$$P(\text{moreinst}(\mathcal{B}, C)|I) = \begin{cases} P(\text{cls}(C)|I) & \text{if } n = 0 \\ P(\text{moreinst}|n) & \text{else} \end{cases}$$

The fourth type of input, an empirical **distribution on the number of object classes** is incorporated similarly.

The final type of input is an **"objectness" measure**, providing the probability $P(\text{obj}(B)|I)$ corresponding to the likelihood of any generic object being contained in box B [1]. We compute the probability of there being an *unannotated* object in the box as the probability of there being *some* object in the box but it is not any of the existing guesses C. Assuming that the likelihood of this box to contain an object is independent of the current set of object detectors, we compute (implicitly conditioning on I):

$$P(\text{newobj}(B)) = P(\text{obj}(B)) \prod_C (1 - P(\text{det}(B, C))) \quad (1)$$

When asked to produce a labeling of the image, the system takes advantage of these computer vision probabilities combined with user input in Section 4.1.2 to return a set of optimal object detections with probabilities $P(\text{det}(B, C)|I, \mathcal{U})$ in Section 4.3.

4.1.2 Incorporating user input

Types of user input. Besides computer vision information, our image labeling system also considers responses from human users. The set $\mathcal{U}$ of user responses consists of five types of *binary* answers: $\mathcal{U}(\text{cls}(C))$ are the responses about object C being in the image, (2) $\mathcal{U}(\text{det}(B, C))$ are the responses about a single instance of object C tightly bound by box B, (3) $\mathcal{U}(\text{moreinst}(\mathcal{B}, C))$ are the responses about set $\mathcal{B}$ covering all instances of class C, (4) $\mathcal{U}(\text{morecls}(\mathcal{C}))$ are the responses about set $\mathcal{C}$ covering all object classes in the image, and (5) $\mathcal{U}(\text{obj}(B))$ are the responses about B containing a single object.

Two things are important to note. First, the goal is to keep the set $\mathcal{U}$ as small as possible: many objects and bounding boxes will not require any user feedback. Second, while the user feedback is incorporated in the model in the form of binary variables, the actual human actions and responses are much more complex (Section 4.2). For example, a user may be asked to draw a bounding box for class C, and then the new box B' is added to the list of boxes under considerations, and $\mathcal{U}(\text{det}(B', C))$ is updated to include a single positive response $u_k = 1$.

Joint computer vision and user input model. To combine computer vision with human input, we adopt the model of [5]. Considering an event E such as "there is an object of class C in the image" or "there are no more instances of class C beyond those in $\mathcal{B}$ boxes," the model estimates the probability this event given image appearance I and the set of user responses $\mathcal{U} = \{u_k\}$ as

$$P(E|I, \mathcal{U}) \propto P(E|I) \prod_k P(u_k|E) \quad (2)$$

This model makes two standard assumptions: first, noise in user responses is independent of image appearance, so $P(\mathcal{U}|I, E) = P(\mathcal{U}|E)$, and second, different user responses are independent from each other, so $P(\mathcal{U}|E) = \prod_k P(u_k|E)$. [5] We learn the true positive and true negative rates $P(u_k = 1|E = 1)$ and $P(u_k = 0|E = 0)$ for each task from humans (Section 5.2).

In contrast to prior approaches, our set of user inputs $\mathcal{U}$ contains multiple types of information, not all of which may be relevant to estimating the probability of every event E. However, each user input u_k affects the probability of multiple events. We provide details in supplementary material.

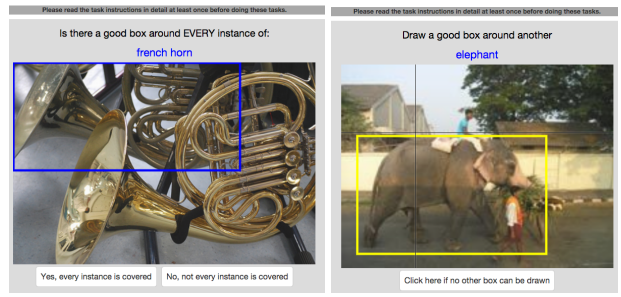

Figure 3. Two of the user interfaces for our human annotation tasks. Other UIs and design details are in supplementary material.

4.2. Human tasks

In our goal to build a human-machine collaboration system for annotating objects in an image, so far we described the strategy for combining computer vision and user input to obtain an image labeling. Now we consider the set of human tasks which the model can utilize to improve this labeling. One important property of our model is that it will automatically find the best question to pose from among the available options, so there's no harm in adding extra tasks.

Table 1 presents the seven tasks we consider. For each question we pose to the users (first column), we can estimate the expected probability of the event being positive using the models of Section 4.1 (second column). Binary feedback from the task is incorporated into the image annotation model in Section 4.1.2 (third column). Figure 3 shows a few of our UIs.

4.3. Optimal annotation

In the process of annotating an image, the system needs to decide which of the human tasks to choose at each step (if any). To do so, we first formalize the requester constraints on precision and utility from Section 3.

Human task	Estimated probability of positive	Update if pos answer / Update if neg answer
Verify-image: does the image contain an object of class C?	$P(\text{cls}(C)\|I,\mathcal{U})$	$1 \rightarrow \mathcal{U}(\text{cls}(C))$ $0 \rightarrow \mathcal{U}(\text{cls}(C))$
Verify-box: is box B tight around an instance of class C?	$P(\text{det}(B,C)\|I,\mathcal{U})$	$1 \rightarrow \mathcal{U}(\text{det}(B,C))$ $0 \rightarrow \mathcal{U}(\text{det}(B,C))$
Verify-cover: are there more instance of class C not covered by the set of boxes $\mathcal{B}$?	$P(\text{moreinst}(\mathcal{B},C)\|I,\mathcal{U})$	$1 \rightarrow \mathcal{U}(\text{moreinst}(\mathcal{B},C))$ $0 \rightarrow \mathcal{U}(\text{moreinst}(\mathcal{B},C))$
Draw-box: draw a new instance of class C not already in set of boxes $\mathcal{B}$.	$P(\text{moreinst}(\mathcal{B},C)\|I,\mathcal{U})$	$1 \rightarrow \mathcal{U}(\text{det}(B',C))$ for drawn box B' $0 \rightarrow \mathcal{U}(\text{moreinst}(\mathcal{B},C))$
Verify-object: is box B tight around *some* object?	$P(\text{obj}(B)\|I,\mathcal{U})$	$1 \rightarrow \mathcal{U}(\text{obj}(B))$ $0 \rightarrow \mathcal{U}(\text{obj}(B)$
Name-box: If box B is tight around an object other than the objects in $\mathcal{C}$, name the object	$P(\text{newobj}(B,\mathcal{C})\|I,\mathcal{U})$	$1 \rightarrow \mathcal{U}(\text{det}(B,C'))$ for named class C' $0 \rightarrow \mathcal{U}(\text{obj}(B))$
Name-image: Name an object in the image besides the known objects $\mathcal{C}$	$P(\text{morecls}\|I,\mathcal{U})$	$1 \rightarrow \mathcal{U}(\text{cls}(C'))$ for named class C' $0 \rightarrow \mathcal{U}(\text{morecls}(\mathcal{C}))$

Table 1. Human annotations tasks and the resulting influence on the annotation model. Details are in Section 4.2.

Consider the object detections $\mathcal{Y} = (B_i, c_i, p_i)\}_{i=1}^N$ of bounding box, class label, and probability tuples, with $p_i = P(\text{det}(B_i, C_i)|I, \mathcal{U})$. Given the provided utility function $f : \mathcal{B} \times \mathcal{C} \rightarrow [0, 1]$ mapping the set of bounding boxes with class labels to how much the user cares about this label, we compute the **expected utility** of a labeling $Y \subseteq \mathcal{Y}$ as

$$\mathbb{E}[\text{Utility}(Y)] = \sum_{i \in Y} p_i f(B_i, C_i) \tag{3}$$

using the linearity of expectation. The simplest case is valuing all detections equally at $f(B, C) = 1 \ \forall B, C$, making utility equal to the number of correct detections.

Similarly, the **expected precision** of labeling $Y \subseteq \mathcal{Y}$ is

$$\mathbb{E}[\text{Precision}(Y)] = \frac{\mathbb{E}[\text{NumCorrect}(Y)]}{N} = \frac{\sum_{i \in Y} p_i}{N} \tag{4}$$

Given the available set $\mathcal{Y}$, the system tries to satisfy the requester constraints. If target precision P^* and utility U^* are requested, the system samples detections from $\mathcal{Y}$ into Y in decreasing order of probability while $\mathbb{E}[\text{Precision}(Y)] \geq P^*$. We define $\text{Precision}(\emptyset) = 1$ so this is always achievable. Since expected utility increases with every additional detection, this will correspond to the highest utility set Y under precision constraint P^*. If $\mathbb{E}[\text{Utility}(Y)] \geq U^*$, the annotation is complete. Otherwise, more human tasks are requested (Section 4.4).

If target precision P^* (or utility U^*) and budget B^* are requested, then we run the annotation system of Section 4.4 until budget is depleted, and produce the set Y as above under the precision constraint P^* (or utility constraint U^*).

4.4. Task selection

The final component of our approach is automatically selecting the right human tasks to best improve the image annotation state. We quantify the tradeoff between cost and accuracy of annotation by formulating it as a Markov decision process (MDP). [28, 10, 44, 27] An MDP consists of states $\mathcal{S}$, actions $\mathcal{A}$, conditional transition probabilities between actions $\mathcal{P}$, and expected rewards of actions $\mathcal{R}$.

States. At each time period of the MDP, the environment is in some state $s \in \mathcal{S}$. In our case, a state S is the set of our current beliefs about the image described in Section 4.1.

Actions. In an MDP, the system takes an action $a \in \mathcal{A}$ from state s, which causes the environment to transition to state s' with probability $\mathcal{P}(s'|s, a)$. In our setting, the set of actions $\mathcal{A}$ correspond to the set of human tasks that the system can request, described in Section 4.2.

Transition probabilities. As a result of an action a from state s, the system move into a new state s', i.e., we update our current understanding of the objects in the image. Transition probabilities correspond to our expectations on the outcomes of the task a. For example, suppose the user is asked to verify if class C is in the image, and suppose our current estimate $P(\text{cls}(C)|I, U) = 0.70$. In Section 5.2 we will learn that average user accuracy rates on this task are $P(u_k = 1|\text{cls}(C) = 1) = 0.87$ and $P(u_k = 0|\text{cls}(C) = 0) = 0.98$. We use Bayes' rule to compute the probability of positive user response as $P(u_k = 1|I, \mathcal{U}) = (0.87)(0.70) + (1 - 0.98)(1 - 0.70) = 0.61$. We can precompute the two resulting states s' (if the user gives a positive answer) and s'' (if the user gives a negative answer), and know that they will occur with probabilities 0.61 and 0.39 respectively if this action a is executed.

Rewards. After transitioning from state s to s' through action a, the agent in an MDP receives a reward with expected value $\mathcal{R}_a(s, s')$. In our case, the expected reward of an action is how much closer the system approached requester specifications on the labeling task. If user has specified the expected precision P^* then at each state s and s' we can compute the maximum number of boxes to return to obtain labelings Y and Y' respectively both with precision at least

Human task	TP	TN	Cost
Verify-image: class C in image?	0.87	0.98	5.34s
Verify-box: class C in box B?	0.77	0.93	5.89s
Verify-cover: more boxes of C?	0.75	0.74	7.57s
Draw-box: draw new box for C	0.72	0.84	10.21s
Verify-object: B *some* object?	0.71	0.96	5.71s
Name-object: name object in B	0.75	0.92	9.67s
Name-image: name object in img	0.98	0.88	9.46s

Table 2. Human annotations tasks with the corresponding accuracy rates and costs. Detailed explanations of each task are in Table 1. TP column is the true positive probability of user answering "yes" (or drawing a box, or writing a name) when the answer should in fact be "yes." For the open-ended tasks we also need to estimate the probabilities of the given answer being *correct*: these probability are draw-box 0.71, name-object 0.94, name-image 0.95. TN column is the true negative probability of the user correctly answering "no." Cost column is median human time in seconds.

P^*. The reward should then be defined as

$$\mathcal{R}_a(s, s') = \frac{\mathbf{E}[Utility(s')] - \mathbf{E}[Utility(s)]}{cost(a)} \quad (5)$$

However, this is quite difficult to optimize in practice since the function is highly discontinuous. Instead we evaluate a continuous estimate of the average expected utility over different levels of precision (similar to average precision over different levels of recall in object detection).[1]

Optimization. Given the transition probabilities and expected rewards, at each step the system chooses the action $a^*(s)$ that maximizes the value $V(s)$, which is computed recursively as

$$a^*(s) = \arg\max_a \left\{ \sum_{s'} P_a(s, s')(R_a(s, s') + V(s')) \right\}$$
$$V(s) = \sum_{s'} P_{a^*(s)}(s, s')(R_{a^*(s)}(s, s') + V(s')) \quad (6)$$

We optimize Equations 6 with 2 steps of lookahead to choose the next action. [10] This is often sufficient in practice and dramatically reduces the computational cost.[2]

5. Experiments

We evaluate both the accuracy and cost of our proposed system which labels objects in images by combining multiple computer vision models with multiple types of human input in a principled framework. We begin by describing the experimental setup (Section 5.1), then discuss the challenges of designing the variety of human tasks and collecting accurate error rates for them (Section 5.2). We then show that having multiple human tasks is important for reducing cost and increasing accuracy of labeling (Section 5.3) and conclude by proving that our system is capable of self-monitoring and able to return labelings according to requester specifications (Section 5.4).

5.1. Setup

We perform experiments on the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) detection dataset [41]. The dataset consists of 400K training images, 20K validation images and 40K test images. The validation and test images are fully annotated with all instances of 200 object classes ranging from accordion to zebra. Since test set annotations are kept hidden by the challenge organizers, we split the validation set into two sets (val1 and val2) and evaluate on val2 following [19].

Computer vision input. We use publicly available code and models as computer vision input. The object detectors are pre-trained RCNN models released by [19]. Image classifiers are convolutional neural network (CNN) classifiers trained with Caffe [26] on ILSVRC2013 detection training set (full images, no bounding box labels) [23]. Detections and classifications with probability less than 0.1 are discarded. [3] The probability distribution for $P(\text{moreinst}|n \text{ inst})$ is computed empirically for all classes jointly on the val1 set. The probability $P(\text{morecls}|n \text{ classes})$ is from [21]. Objectness probabilities are extracted using the code of [1].

Human-computer interaction. Setting up a system that integrates computer vision knowledge with human input requires finding common ground between the two. One necessary decision is what bounding box is considered a correct detection. In object detection, a bounding box is commonly considered correct if its intersection over union (IOU) with a ground truth box is greater than 0.5. [41, 16] However, training humans to visually inspect a bounding box with IOU of 0.3 and distinguish it from one with IOU 0.5 is surprisingly difficult (Figure 4). In our experiments we choose 0.7 as the target IOU as the halfway point between the targets of object detection and human annotation.[4]

The higher IOU further reduces the accuracy of automated object detection, from 34.1% mAP with IOU of 0.5 and non-maximum suppression (nms) of 0.3 as in [19] to 18.7% mAP with IOU of 0.7 and nms of 0.5.

5.2. Learning human accuracy rates

To incorporate human input into our image labeling system, we need to compute the expected outcome of an action

[1] We treat budget constraints as rigid, so $\mathcal{R}_a(s, s') = -\inf$ if the cost of action a is less than the remaining budget.

[2] Note that doing 1 step of lookahead is not sufficient because some tasks in Table 1 do not directly influence the labeling.

[3] Details about probability calibration are in supplementary material.

[4] When human annotators are used to collect object detection datasets, the average difference in bounding boxes for the same instance between two annotators is about 5 pixels on each side. [41] For an 200x200 pixel object, this corresponds to approximately 0.90 intersection over union.

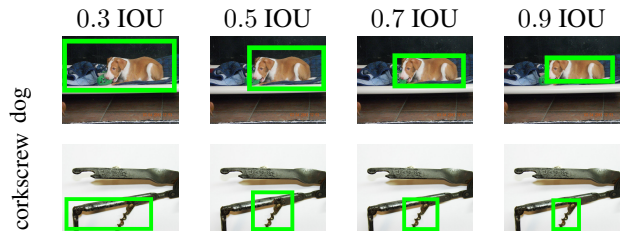

Figure 4. Bounding boxes with increasing intersection over union (IOU) with the optimal tight box. Training human annotators to make binary decision on whether or not a bounding box is a good detection is quite difficult; this the primary contributor to human error rates. Guidance such as "the object occupies more than half the bounding box" is confusing since objects like corkscrews (bottom row) occupy a small area even at perfect IOU.

(Section 4.2). To do so, we collect user accuracy rates for each human task of Table 1. We assume that user error is dependent only on the type of task (for example, on the clarity of instructions or the effectiveness of filtering spam workers) and not on the exact question: i.e., a user is equally likely to misclassify a cat as she is to misclassify a hammer.

To compute the error rates, we need a positive and negative set of examples for each task to estimate the true positive and true negative rate of user responses. In most cases we generated these sets directly from ILSVRC val1 images and annotations.[5] We showed the generated positive and negative sets of questions for each task to workers on AMT. Quality control was done by planting a few "gold standard" questions and preventing users from submitting if they didn't achieve expected accuracy on those questions.

The accuracy rates and costs (in median human time [13]) are reported in Table 2. By far the biggest source of error is getting users to make binary decisions on tasks with a bounding box: the average accuracy is 0.92 for image-level tasks (verify-image and name-image) and 0.81 for the box-level tasks. For the open-ended tasks (draw-box, name-object, name-image) we needed to compute both the probability that the user *answers* the question affirmatively (i.e., attempts to draw a box) as well as as the probability that the user is *correct*. For name-object and name-image we manually verified the responses on 100 images each. Some common mistakes were misclassifications (calling a sheep "goat" or a cello "violin") and annotations that were too general (e.g., "food") despite instructions.

5.3. Evaluating multiple sources of input

We evaluate our system on 2216 images of val2 that contain at least 4 ground truth object instances using the human accuracy rates collected in Section 5.2 to simulate the real-world labeling scenario. Incorporating computer vision and human tasks into one coherent object annotation framework provides significant improvements over using computer vision or human input in isolation. Further, we show that multiple types of human tasks are necessary for optimal benefit.

Figure 6(a) shows the average utility of a labeling as a function of budget (human labeling time). The utility function is defined as the number of correct detections in this case. Average utility is computed by averaging across multiple levels of precision on each image and then across all images in the dataset. For the purposes of simulation, since only the 200 object categories in the image are known, we omit the verify-object and name-object tasks.

The object detectors alone have average utility of 0.95 objects per image. Adding in just the verify-image and verify-box tasks improves the utility 1.5x after 4.5 minutes of labeling. In contrast, a system with all human tasks achieves this utility after 45 seconds. Further, the full system (containing both computer vision and all human tasks) continues to improve in utility over time, obtaining 5.98 average utility after 600 seconds of labeling: 6.2x higher than object detection alone. The draw-box tasks helps correct missed detections, and write-name tasks corrects missed image classifications. Removing the write-name task reduces the utility slightly to 5.51 at 10 minutes of labeling.

The importance of computer vision input is apparent early in the annotation process. After 30 seconds of annotation, the computer+human method (with all tasks) outperforms the utility of the human-only method (with all tasks) by 1.92x. This means that given 30 seconds of human annotation time per image, adding in computer vision input can almost double the accuracy of the human labeling. However, given a budget of 5 minutes the benefits of computer vision will become less significant and the human-only method will perform comparably to the joint method.

Figure 5 shows results of our labeling system.

5.4. Respecting requester constraints

One of the key aspects of our system is the ability to allow the requester to provide constraints on the desired annotation (Section 3). After the annotation process (600 seconds), we queried the system for image annotations at 0.5 precision; 0.519 of the returned objects were indeed correct detections. We repeated the process at 0.1 intervals up to 0.9 precision; the model returned detections with an average of 0.041 higher precision than queried. Thus, the system is well-calibrated and we can do requester queries.

Figure 6(b) plots requested budget (x-axis) and requested precision (colored lines) versus the utility of returned labeling. We observe that, given the same budget, requesting a higher level of precision causes the system to be more cautious about returned detections and thus results in lower-utility labelings.After incorporating 5 minutes of human input and requesting a labeling at 0.9 precision the system will return on average 4 correctly labeled objects.

[5]Details about generating these sets are in supplementary material.

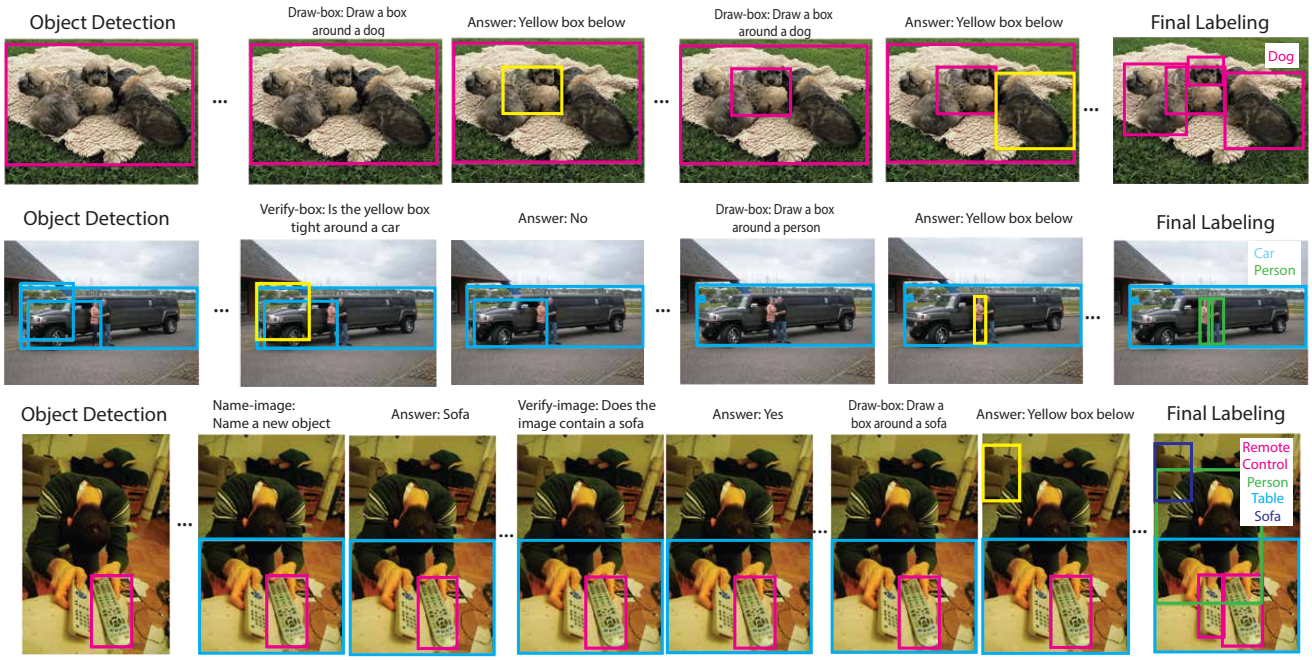

Figure 5. Some example results from our system integrating computer vision with multiple types of user feedback to annotate objects.

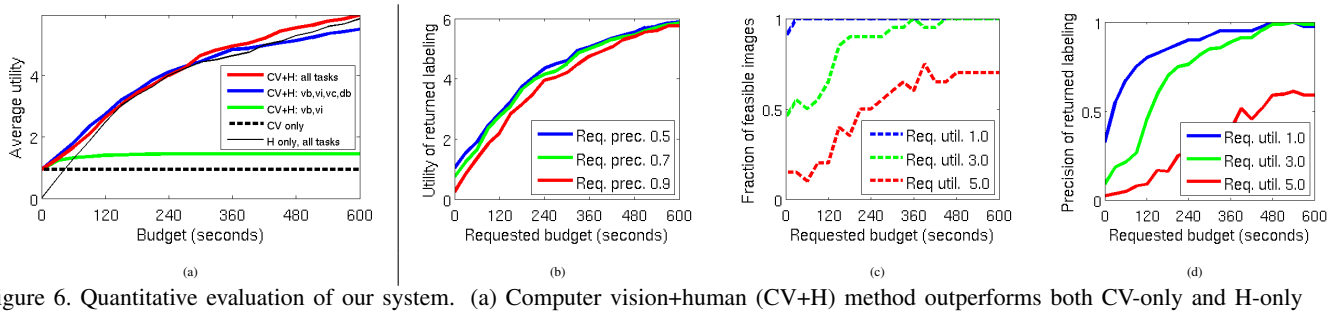

Figure 6. Quantitative evaluation of our system. (a) Computer vision+human (CV+H) method outperforms both CV-only and H-only methods; details in Section 5.3. (b-d) Quality of returned labeling as a function of requester constraints; details in Section 5.4.

Instead of specifying the desired budget and precision, the requester can also specify the desired budget and utility. However, this may not be feasible in all cases, as shown in Figure 6(c). For example, obtaining a utility of 3 objects labeled after 60 seconds of labeling is feasible in only 50% of the images. For the images where it is feasible, however, we can refer to Figure 6(d) to get the expected precision of the labeling. In this case, the expected precision is 21.2%.

Providing this detailed analysis of the tradeoff between precision, utility and budget can help requesters interested in obtaining a dense labeling of objects in an image a-priori estimate the quality of the labeling given the scope of their problem and their constraints.

6. Conclusions

We presented a principled approach to unifying multiple inputs from both computer vision and humans to label objects in images. We conclude with several take-home messages. First, from the **computer vision perspective**, object detectors are considered to have correctly detected an object if they return a loose bounding box (IOU 0.5) and accuracy drops rapidly when a tighter bounding box is required. However, this is problematic for higher-level applications (such as integrating detectors with human feedback).

From a **crowd engineering perspective**, we demonstrated that it is worthwhile to combine multiple tasks in a principled framework. One interesting observation is that the verify-cover task (asking if all instances of an object class are already labeled in the image) inspired by ILSVRC data collection process [41] turned out in practice to have almost no impact on the labeling accuracy as it was selected by the model less than 0.1% of the time. This confirmed more of the intuitions of the later COCO [35] dataset that asking slightly more complex human tasks (such as putting a dot on the object rather than merely asking if the object in the image, or drawing the bounding box around an unanno-

tated instance rather than merely asking if one exists) may be more efficient.

Finally, from an **application developer perspective**, we show that even though computer vision is not yet ready to detect all objects, we have a principled way of labeling all objects in a scene, trading off precision, utility and budget.

References

[1] B. Alexe, T. Deselares, and V. Ferrari. Measuring the objectness of image windows. In *PAMI*, 2012. 2, 4, 6

[2] A. C. Berg, T. L. Berg, H. Daume, J. Dodge, A. Goyal, X. Han, A. Mensch, M. Mitchell, A. Sood, K. Stratos, et al. Understanding and predicting importance in images. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 3562–3569. IEEE, 2012. 2

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

[4] J. Bragg, Mausam, and D. S. Weld. Crowdsourcing multi-label classification for taxonomy creation. In *HCOMP'13*, 2013. 2

[5] S. Branson, C. Wah, B. Babenko, F. Schroff, P. Welinder, P. Perona, and S. Belongie. Visual recognition with humans in the loop. In *ECCV*, 2010. 1, 2, 4

[6] W. C., B. S., P. P., and B. S. Multiclass recognition and part localization with humans in the loop. In *IEEE International Conference on Computer Vision (ICCV)*, 2011. 1, 2

[7] Q. Chen, Z. Song, Z. Huang, Y. Hua, and S. Yan. Contextualizing object detection and classification. volume PP, 2014. 2

[8] X. Chen, A. Shrivastava, and A. Gupta. NEIL: Extracting Visual Knowledge from Web Data. In *International Conference on Computer Vision (ICCV)*, 2013. 1, 2

[9] S. Cooper, F. Khatib, A. Treuille, J. Barbero, J. Lee, M. Beenen, A. Leaver-Fay, D. Baker, and Z. Popovic. Predicting protein structures with a multiplayer online game. *Nature*, 466:756—760, 2010. 1

[10] P. Dai, D. S. Weld, et al. Decision-theoretic control of crowdsourced workflows. In *Twenty-Fourth AAAI Conference on Artificial Intelligence*, 2010. 2, 5, 6

[11] T. Dean, M. Ruzon, M. Segal, J. Shlens, S. Vijayanarasimhan, and J. Yagnik. Fast, accurate detection of 100,000 object classes on a single machine. In *CVPR*, 2013. 1, 2

[12] J. Deng, J. Krause, and L. Fei-Fei. Fine-grained crowdsourcing for fine-grained recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2013. 2

[13] J. Deng, O. Russakovsky, J. Krause, M. Bernstein, A. C. Berg, and L. Fei-Fei. Scalable multi-label annotation. In *CHI*, 2014. 2, 7

[14] C. Desai, D. Ramanan, and C. Fowlkes. Discriminative models for multi-class object layout. *International Journal of Computer Vision (IJCV)*, 2011. 1

[15] S. Divvala, A. Farhadi, and C. Guestrin. Learning everything about anything: Webly-supervised visual concept learning. 2014. 1, 2

[16] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The Pascal Visual Object Classes (VOC) challenge. *IJCV*, 88(2):303–338, June 2010. 1, 6

[17] P. Felzenszwalb, R. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part based models. *PAMI*, 32, 2010. 3

[18] H. Gao, G. Barbier, and R. Goolsby. Harnessing the crowdsourcing power of social media for disaster relief. *IEEE Intelligent Systems*, 26(3):10–14, 2011. 1

[19] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014. 6

[20] R. B. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation (v4). *CoRR*, 2013. 1, 2, 3

[21] M. R. Greene. Statistics of high-level scene context. *Frontiers in Psychology*, 4, 2013. 1, 2, 6

[22] G. Hartmann, M. Grundmann, J. Hoffman, D. Tsai, V. Kwatra, O. Madani, S. Vijayanarasimhan, I. Essa, J. Rehg, and R. Sukthankar. Weakly supervised learning of object segmentations from web-scale video. In *ECCV Workshop on Web-scale Vision and Social Media*, 2012. 2

[23] J. Hoffman, S. Guadarrama, E. Tzeng, J. Donahue, R. Girshick, T. Darrell, and K. Saenko. LSDA: Large scale detection through adaptation. arXiv:1407.5035, 2014. 1, 2, 3, 6

[24] S. J. Hwang and K. Grauman. Reading between the lines: Object localization using implicit cues from image tags. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 34(6):1145–1158, 2012. 2

[25] S. D. Jain and K. Grauman. Predicting sufficient annotation strength for interactive foreground segmentation. December 2013. 1, 2

[26] Y. Jia. Caffe: An open source convolutional architecture for fast feature embedding. http://caffe.berkeleyvision.org/, 2013. 6

[27] E. Kamar, S. Hacker, and E. Horvitz. Combining human and machine intelligence in large-scale crowdsourcing. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems - Volume 1*, AAMAS '12, 2012. 2, 5

[28] S. Karayev, M. Fritz, and T. Darrell. Anytime recognition of objects and scenes. In *CVPR*, 2014. 5

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

[30] A. Kovashka, S. Vijayanarasimhan, and K. Grauman. Actively selecting annotations among objects and attributes. *ICCV*, 2011. 2

[31] A. Krizhevsky, I. Sutskever, and G. Hinton. ImageNet classification with deep convolutional neural networks. In *NIPS*, 2012. 2, 3

[32] S. Lad and D. Parikh. Interactively guiding semi-supervised clustering via attribute-based explanations. In *Computer Vision–ECCV 2014*, pages 333–349. Springer, 2014. 1, 2

[33] C. Li, D. Parikh, and T. Chen. Automatic discovery of groups of objects for scene understanding. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 2735–2742. IEEE, 2012. 1

[34] L.-J. Li, R. Socher, and L. Fei-Fei. Towards total scene understanding:classification, annotation and segmentation in an automatic framework. In *CVPR*, 2009. 1

[35] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollr, and C. L. Zitnick. Microsoft COCO: Common Objects in Context. In *ECCV*, 2014. 1, 2, 8

[36] C. J. Lintott, K. Schawinski, A. Slosar, K. Land, S. Bamford, D. Thomas, M. J. Raddick, R. C. Nichol, A. Szalay, D. Andreescu, P. Murray, and J. Vandenberg. Galaxy zoo: morphologies derived from visual inspection of galaxies from the sloan digital sky survey. *MNRAS*, 389(3):1179–1189, 2008. 1

[37] D. Parikh and K. Grauman. Relative attributes. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 503–510. IEEE, 2011. 1

[38] A. Parkash and D. Parikh. Attributes for classifier feedback. In *European Conference on Computer Vision (ECCV)*, 2012. 1, 2

[39] J. Peng, Q. Liu, A. Ihler, and B. Berger. Crowdsourcing for structured labeling with applications to protein folding. In *Proc. ICML Machine Learning Meets Crowdsourcing Workshop*, 2013. 1

[40] A. Prest, C. Leistner, J. Civera, C. Schmid, and V. Ferrari. Learning object class detectors fromweakly annotated video. In *Computer Vision and Pattern Recognition (CVPR)*, 2012. 2

[41] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. Imagenet large scale visual recognition challenge, 2014. 1, 2, 6, 8

[42] M. A. Sadeghi and A. Farhadi. Recognition using visual phrases. 2011. 1

[43] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. *CoRR*, abs/1312.6229, 2013. 1

[44] V. S. Sheng, F. Provost, and P. G. Ipeirotis. Get another label? Improving data quality and data mining using multiple, noisy labelers. In *SIGKDD*, 2008. 2, 5

[45] Z. Song, Q. Chen, Z. Huang, Y. Hua, and S. Yan. Contextualizing object detection and classification. In *CVPR*, 2011. 1

[46] M. Spain and P. Perona. Some objects are more equal than others: Measuring and predicting importance. In *Proceedings of the 10th European Conference on Computer Vision: Part I*, ECCV '08, pages 523–536, Berlin, Heidelberg, 2008. Springer-Verlag. 2

[47] H. Su, J. Deng, and L. Fei-Fei. Crowdsourcing annotations for visual object detection. In *AAAI Human Computation Workshop*, 2012. 1, 2

[48] K. Tang, R. Sukthankar, J. Yagnik, and L. Fei-Fei. Discriminative segment annotation in weakly labeled video. In *CVPR*, 2013. 2

[49] K. E. A. van de Sande, C. G. M. Snoek, and A. W. M. Smeulders. Fisher and vlad with flair. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2014. 1

[50] S. Vijayanarasimhan and K. Grauman. Multi-level active prediction of useful image annotations for recognition. In *NIPS*, 2009. 2

[51] S. Vijayanarasimhan and K. Grauman. Predicting effort vs. informativeness for multi-label image annotations. In *CVPR*, 2009. 2

[52] S. Vijayanarasimhan and K. Grauman. Large-scale live active learning: Training object detectors with crawled data and crowds. *International Journal of Computer Vision*, 108(1-2), 2014. 2

[53] L. von Ahn and L. Dabbish. Esp: Labeling images with a computer game. In *AAAI Spring Symposium: Knowledge Collection from Volunteer Contributors*, 2005. 2

[54] C. Vondrick, D. Patterson, and D. Ramanan. Efficiently scaling up crowdsourced video annotation. *International Journal of Computer Vision*, 2013. 1, 2

[55] C. Wah, G. V. Horn, S. Branson, S. Maji, P. Perona, and S. Belongie. Similarity comparisons for interactive fine-grained categorization. In *Computer Vision and Pattern Recognition (CVPR)*, Columbus, OH, June 2014. 1, 2

[56] X. Wang, M. Yang, S. Zhu, and Y. Lin. Regionlets for generic object detection. In *ICCV*, 2013. 1

[57] D. S. Weld, P. Dai, et al. Human intelligence needs artificial intelligence. In *Workshops at the Twenty-Fifth AAAI Conference on Artificial Intelligence*, 2011. 2

[58] C. Wojek, S. Walk, S. Roth, K. Schindler, and B. Schiele. Monocular visual scene understanding: Understanding multi-object traffic scenes. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 2013. 1

[59] J. Xiao, J. Hays, K. Ehinger, A. Oliva, and A. Torralba. SUN database: Large-scale scene recognition from Abbey to Zoo. *CVPR*, 2010. 1, 3

[60] P. Zhang, J. Wang, A. Farhadi, M. Hebert, and D. Parikh. Predicting failures of vision systems. In *CVPR*, 2014. 2