Best of both worlds: human-machine collaboration for object annotation

Olga Russakovsky[1] Li-Jia Li[2] Li Fei-Fei[1]
[1]Stanford University [2]Snapchat*

Abstract

The long-standing goal of localizing every object in an image remains elusive. Manually annotating objects is quite expensive despite crowd engineering innovations. Current state-of-the-art automatic object detectors can accurately detect at most a few objects per image. This paper brings together the latest advancements in object detection and in crowd engineering into a principled framework for accurately and efficiently localizing objects in images. The input to the system is an image to annotate and a set of annotation constraints: desired precision, utility and/or human cost of the labeling. The output is a set of object annotations, informed by human feedback and computer vision. Our model seamlessly integrates multiple computer vision models with multiple sources of human input in a Markov Decision Process. We empirically validate the effectiveness of our human-in-the-loop labeling approach on the ILSVRC2014 object detection dataset.

1. Introduction

The field of large-scale object detection has leaped forward in the past few years [20, 43, 11, 45, 61, 24, 53], with significant progress both in techniques [20, 43, 53, 45] 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 [43, 16], and achievements such as 1.9x improvement in accuracy between year 2013 and 2014 on the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [43] 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 [36, 63, 35] or for modeling object context [64, 14, 44, 48]. However, this is still currently not enough to go from detecting 6 to detecting 100 objects.

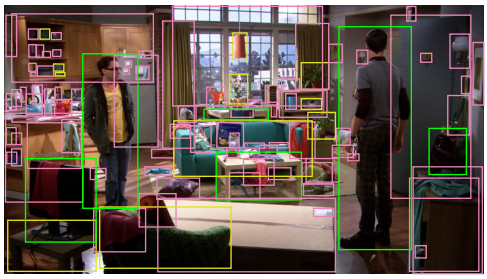

Figure 1. This cluttered image has 100 annotated objects shown with green, yellow and pink boxes. The green boxes correspond to the 6 objects correctly detected by the state-of-the-art RCNN model [20] trained on the ILSVRC dataset [43]. (The about 500 false positive detections are not shown.) Yellow 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.

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 [41, 9], disaster relief distribution [18] and galaxy discovery [38]. In computer vision with human-in-the-loop approaches, human intervention has ranged from binary question-and-answer [6, 59, 60] to attribute-based feedback [40, 39, 34] to free-form object annotation [58]. For understanding all objects in an image, one important decision is which questions to pose to humans. Binary questions are not sufficient. Asking humans to draw bounding boxes is expensive: obtaining an accurate box around a single object takes between 7 seconds [26] to 42 seconds [50], and with 23 objects in an average indoor scene [22] the costs quickly add up. Based on insights from object detection dataset construction [37, 43], it is best to use a variety of human interventions; however, trading off accuracy and cost of annotation becomes a challenge.

We develop a principled framework integrating state-of-the-art scene understanding models [20, 33, 1, 22] with

*This work was done while Li-Jia Li was at Yahoo! Labs

state-of-the-art crowd engineering techniques [43, 37, 10, 46, 28] 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 in both. [6, 26] One key component, in contrast to prior work, is the incorporation of feedback from multiple types of human input and from multiple computer vision models.

2. **Automatically trades off density, precision and cost of annotation** in a principled framework.

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

4. **Is light-weight and easily extensible**. The framework is able to continuously incorporate the latest computer vision and crowd engineering innovations.

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

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 [6, 59, 12, 60], image segmentation [26], attribute-based classification [32, 40, 3], image clustering [34], image annotation [54, 55, 47], and human interaction [31] and object annotation in videos [58]. Methods such as [6, 59, 12, 60] jointly model human and computer uncertainty and characterize human time versus annotation accuracy, but only incorporate a single type of human response. Works such as [26, 13, 54] use multiple modalities of human feedback, with varying costs, and accurately model and predict the success of each modality. However, they do not incorporate iterative improvement in annotation. We build upon these approaches to integrate multiple human annotation modalities with state-of-the-art computer vision models in an iterative framework for the challenging object annotation task.

Better object detection. Methods have been developed for training better object detection models with weakly supervised data [42, 23, 52, 8, 24, 15]. Active learning approaches has been developed to improve object detectors with minimal human annotation cost during training [32, 56]. 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 [57, 12, 30], tricks to reduce the annotation search space [13, 4], more effective user interface design [50, 58], making use of existing annotations [5], making use of weak human supervision [26, 7] and accurately computing the number of required workers [46]. These innovations are important in our framework for minimizing the cost of human annotation when it is needed to augment computer vision. Approaches such as [10, 46, 28, 62] use iterative improvement to perform a task with accuracy per unit of human cost. 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. The input to the system is an image to annotate and a set of annotation constraints. The output is a set of bounding box annotations with object names. 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 annotation tasks).

The requester may specify up to two of three constraints:

1. **Utility.** In the simplest case, utility of a labeling corresponds to the number of objects. However, since some objects in the image may be more important than others [49, 25, 2], the requester may optionally specify a function mapping each image region and class label to a real value indicating importance. The requester then specifies the minimum total utility of the labels.

2. **Precision.** When the system returns N bounding box annotations with object names, if N_C of them are correct detections, then precision is $\frac{N_C}{N}$. The requester can specify the minimum required level of precision.

3. **Budget.** In our formulation, budget corresponds to cost of human time although methods such as [58] can be applied to also incorporate CPU cost.

On one end of the spectrum the requester can set the maximum budget to zero, and obtain the best automatic 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 detailed annotation similar to that of the SUN dataset [64].

4. Method

The system uses both computer vision and user input to annotate objects in images subject to provided constraints

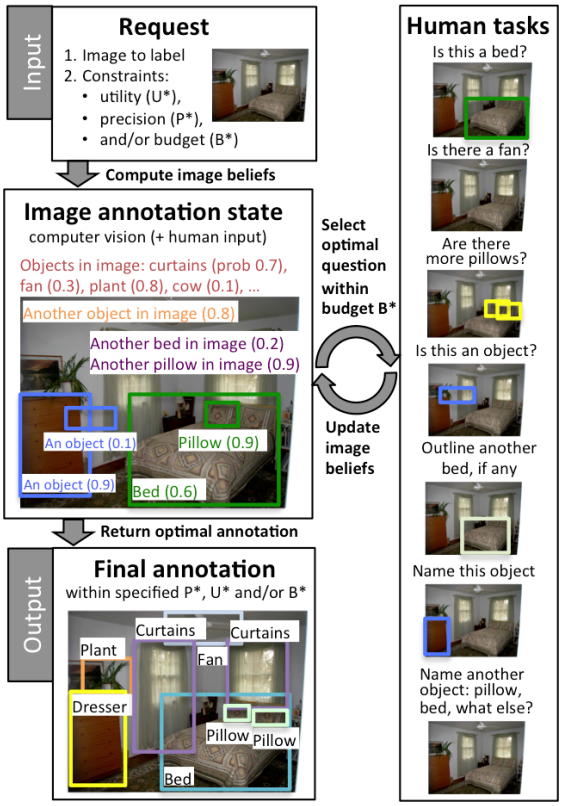

Figure 2. Overview of our system. Given a request for annotating an image, the system alternates between updating the image annotation and soliciting user feedback through human tasks. Upon satisfying the requester constraints, it terminates and returns a image with a set of bounding box annotations.

(Figure 2. It alternates between getting user feedback and updating the image probabilities. Section 4.1 formalizes the requester constraints. Section 4.2 presents the core of our system: the selection of optimal human questions. Secton 4.3 describes the probabilistic framework for combining computer vision with human input, with Section 4.3.1 focusing on the computer vision models.

4.1. Annotation evaluation

Let $\mathcal{Y} = (B_i, C_i, p_i)\}_{i=1}^N$ be the set of N object detections, each with bounding box B_i, class label c_i, and probability of detection being correct p_i. We now explain how our human-in-the-loop system evaluates $\mathcal{Y}$ and outputs the final annotation according to requester constraints.

The **expected precision** of any labeling $Y \subseteq \mathcal{Y}$ is

$$\mathbb{E}[\text{Precision}(Y)] = \frac{\mathbb{E}[\text{NumCorrect}(Y)]}{|Y|} = \frac{\sum_{i \in Y} p_i}{|Y|} \quad (1)$$

using the linearity of expectation. Similarly, given the requester provided utility function $f : \mathcal{B} \times \mathcal{C} \to [0, 1]$ mapping the set of bounding boxes with class labels to how much the requester cares about this label, the **expected utility** is

$$\mathbb{E}[\text{Utility}(Y)] = \sum_{i \in Y} p_i f(B_i, C_i) \quad (2)$$

The simplest case (used in this paper) is valuing all detections equally at $f(B, C) = 1 \; \forall B, C$, making utility equal to the number of correct detections.

Annotation given constraints. Given the available set $\mathcal{Y}$, the system tries to output a labeling Y that satisfies the requester constraints. Recall that requester specified at most two of the three constraints of utility, precision and budget. If both target utility U^* and precision P^* 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 constraints are satisfied. If not, we continue the labeling system.

If target precision P^* (or utility U^*) and budget B^* are specified, then we run the annotation system of Section 4.2 until budget is depleted, and produce the set Y as above under the precision constraint P^* (or utility constraint U^*).

Approximation of annotation quality. As the annotation system progresses, it needs to evaluate the quality of annotation set $\mathcal{Y}$. One option is to directly evaluate how closely $\mathcal{Y}$ satisfies requester constraints: for example, by producing the set $Y \subseteq \mathcal{Y}$ which satisfies the requested level of precision P^* and using $\mathbb{E}[\text{Utility}(Y)]$ as the objective. However, this measure is discontinuous and difficult to optimize. Since precision and utility are closely related, we directly use $\mathbb{E}[\text{Utility}(\mathcal{Y})]$ as the objective.

4.2. MDP formulation for human task selection

The main component of our approach is automatically selecting the right human question 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). [29, 10, 46, 28, 21] An MDP consists of states $\mathcal{S}$, actions $\mathcal{A}$, conditional transition probabilities $\mathcal{P}$, 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 our set of current beliefs about the image I, computed by combining computer vision models with user input. For simplicity, in this work we don't update the computer vision models as annotation progresses on a single image, so the only dynamic part of $\mathcal{S}$ in the user input U.

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 questions that

Human tasks (MDP actions)
Verify-box: is box B tight around an instance of class C?
Verify-image: does the image contain an object of class C?
Verify-cover: are there more instance of class C not covered by the set of boxes $\mathcal{B}$?
Draw-box: draw a new instance of class C not already in set of boxes $\mathcal{B}$.
Name-image: Name an object class in the image besides the known object classes $\mathcal{C}$
Verify-object: is box B tight around *some* object?
Name-box: If box B is tight around an object other than the objects in $\mathcal{C}_B$, name the object

Table 1. Human annotations tasks. One important property of our model is that it will automatically find the best question to pose, so there's no harm in adding extra tasks.

the system can ask. The types of human tasks are listed in Table 1. Each question is one of the tasks grounded to the image: for example, "verify-box: is box at $(10, 50, 37, 89)$ an instance of class cat?" or "draw-box: draw a box around another instance of table besides $(83, 119, 74, 281)$ and $(281, 470, 46, 24)$". Figure 3 shows some example UIs.

Transition probabilities. As a result of an action a from state s, the system moves into a new state s'; in other words, the current beliefs about the image get updated by the addition of a new user response u_t to $\mathcal{U}$. Transition probabilities correspond to our expectations on the outcome (user response) of the question a (Section 4.3).

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 states contain object detection annotations $\mathcal{Y}(s)$ and $\mathcal{Y}(s')$ respectively (Section 4.3) with detection probabilities computed in Section 4.3. Using the definition of Section 4.1, the reward is

$$\mathcal{R}_a(s, s') = \frac{\mathbb{E}[\text{Utility}(\mathcal{Y}(s'))] - \mathbb{E}[\text{Utility}(\mathcal{Y}(s))]}{cost(a)} \quad (3)$$

We treat budget constraint as rigid, so $\mathcal{R}_a(s, s') = -\inf$ if the $cost(a)$ is less than the remaining budget. The system terminates once $\mathcal{Y}(s)$ satisfies the requester constraints.

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

$$a^*(s) = \arg\max_a \left\{ \sum_{s'} P_a(s, s')(R_a(s, s') + V(s')) \right\}$$
$$V(s) = \sum_{s'} P_{a^*(s)}(s, s')(R_{a^*(s)}(s, s') + V(s')) \quad (4)$$

We optimize Equations 4 with 2 steps of lookahead to choose the next action. [10] This is often sufficient in practice and dramatically reduces the computational cost.[1]

[1] Doing 1 step of lookahead is not sufficient because some tasks in Table 1 (e.g., name-image) do not directly influence the labeling.

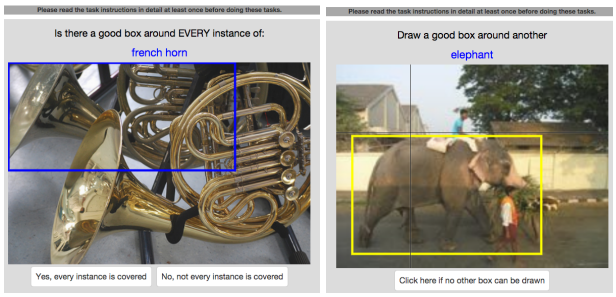

Figure 3. Two of the user interfaces for our human annotation tasks. Other UIs and design details are in supplementary material.

4.3. Human-in-the-loop probabilistic framework

Our system is based on combining computer vision with human input into one probabilistic framework. Consider the MDP on image I at time step T, after $T-1$ actions $a_1 \dots a_{T-1}$ were taken and user responses $\mathcal{U}_{T-1} = \{u_t\}_{t=1}^{T-1}$ obtained. We need to compute two closely related quantities: MDP transition probabilities and object detection probabilities for the labeling. We begin by describing the former, and then show how the latter is a special case.

MDP transition probabilities. The MDP is now in state s and we set out to compute the transition probabilities to new states. The next state is uniquely determined by the action (i.e., question) a_T that the system chooses to ask and by the user response u_T. Thus for each a_T we need to compute the probability of each response u_T given the image I and user responses so far $\mathcal{U}_{T-1}$. Let $E_1^T \dots E_K^T$ be the set of possible answers to this question a_T. By law of total probability and Bayes' rule:

$$P(u_T|I, \mathcal{U}_{T-1}) = \sum_{k=1}^{K} P(u_T|E_k^T) P(E_k^T|I, \mathcal{U}_{T-1}) \quad (5)$$

The first term of Eqn. 5 is $P(u_T|E_k^T)$, which corresponds to the probability of user giving an answer u_T if E_k^T were the correct answer to a_T.[2] We simplified this term from $P(u_T|E_k^T, I, \mathcal{U}_{T-1})$ by making two assumptions following [6]: (1) noise in user responses is independent of image appearance I if correct answer E_k^T is given, and (2) user responses are independent of each other. To compute $P(u_T|E_k^T)$ we will use the empirical error rates of Section 5.2 for each question type in Table 1.

The second term of Eqn. 5 is $P(E_k^T|I, \mathcal{U}_{T-1})$, which corresponds to the probability of answer E_k^T to a_T in fact

[2] There are only a few possible answers to each question which allows us to enumerate them. For example, if the user is asked to perform the draw-box task at time T, the only possible answers are E_1^T: the user draws a box, or E_2^T : the user stated that no box can be drawn.

being correct. Applying Bayes' rule again:

$$P(E_k^T|I,\mathcal{U}_{T-1}) \propto P(E_k^T|I) \prod_{t=1}^{T-1} P(u_t|E_k^T,I,\mathcal{U}_{t-1}) \quad (6)$$

Here, $P(E_k^T|I)$ is the computer vision model for E_k^T (described in Section 4.3.1). $P(u_t|E_k^T,I,\mathcal{U}_{t-1})$ is very similar to the first term in Eqn. 5 with one important exception: it unifies user response u_t to action a_t at time t with potential answer E_k^T to action a_T at time T. We consider two cases.

Case 1: The correct response to question a_t at time t can be inferred from E_k^T at time T. For example, suppose a_t is "is there an object of class c_i in the image?" and E_k^T is "box B_i is tight around an instance of class c_i." Then the correct response "yes" to a_t can be inferred from E_k^T. The supplementary material provides a complete list of these relationships. Let the inferred correct answer be E_m^t. In this case, we again apply the model of [6] to simplify $P(u_t|E_k^T,I,\mathcal{U}_{t-1}) = P(u_t|E_m^t)$ as above.

Case 2: The correct response to question a_t at time t is independent of E_k^T at time T. For example, suppose a_t is as above "is there an object of class c_i in the image?" but E_k^T is "box B_j is tight around an instance of class c_j." Since E_k^T does not provide any information regarding the correct response to a_t, we know $P(u_t|E,I,\mathcal{U}_{t-1}) = P(u_t|I,\mathcal{U}_{t-1})$.[3] To compute this we apply Eqn. 5.

This concludes the transition probability computation for every action a_T and every possible user response u_T. New actions can seamlessly be added to the MDP framework assuming the set of possible answers $E_1,\dots E_k$, the computer vision probabilities $P(E_k|I)$ and the user error probabilities $P(u|E_k)$ can be computed for each new action type.

Object detection probabilities. In addition to transition probabilities, we also need to compute the object detection probabilities to be used in the image labeling. For a detection with bounding box B and class label c, let $\hat{E}$ be "B is a tight box around an instance of class c." The probability of the detection being correct given the information available at time T is $P(\hat{E}|I,\mathcal{U}_{T-1})$. This is computed directly with Eqn. 6, with $P(\hat{E}|I)$ from an object detector.

One extra consideration is that (B,c) can be automatically proposed by the object detector or manually by the users.[4] If the box B was manually drawn at some previous time $\hat{T}$ in response to the draw-box task, then we omit the computer vision model and modify Eqn. 6 slightly to

$$P(\hat{E}|I,\mathcal{U}_{T-1}) \propto P(\hat{E}|u_{\hat{T}}) \prod_{t} P(u_t|\hat{E},I,\mathcal{U}_{t-1}) \quad (7)$$

with the product ranging over $t \in \{1,\dots,T-1\} - \hat{T}$. $P(\hat{E}|u_{\hat{T}})$ is the empirical user accuracy rate for the drawn bounding box to actually be *correct* (Section 5.2).

[3] Alternatively, our model can be extended to include object-object co-occurrence information here.

[4] We use the same reasoning for name-image and name-box tasks.

4.3.1 Computer vision input

We incorporate multiple computer vision models into our system to compute the above transition probabilities for all actions of Table 1:

(1) Detection. Computing transition probabilities corresponding to the *verify-box* action with box B and class C requires computing $P(\text{det}(B,C)|I)$: the probability that B is tight around an instance of class C on image I. Standard object detectors can be used here e.g., [20, 17, 24].

(2) Classification. Similarly, the *verify-image* action for object class C require computing $P(\text{cls}(C)|I)$ that C is present in the image. Models such as [33, 51] can be used.

(3) Another instance. Computing transition probabilities for *verify-cover* and *draw-box* actions for class C and set of boxes $\mathcal{B}$ require computing $P(\text{more}(\mathcal{B},C)|I)$ that there are other instances of C in the image beyond those contained in $\mathcal{B}$. We compute this probability using an empirical *distribution on number of object instances* in images. It provides the probability $P(\text{more}|n)$ of there being more instances of an object class given the image is known to contain at least n instances of this class. Let $\text{nc}(\mathcal{B},C)$ be the number of boxes $\mathcal{B}$ that are correct for class C, 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

$$P(\text{more}(\mathcal{B},C)|I) = \begin{cases} P(\text{cls}(C)|I) & \text{if } n=0 \\ P(\text{more}|n) & \text{else} \end{cases} \quad (8)$$

(4) Another class. Similarly, a *name-image* action requires computing $P(\text{morecls}(\mathcal{C}|I))$ that another object class is present in the image beyond the classes $\mathcal{C}$. An empirical *distribution on number of object classes* is used as above.

(5) Objectness. A *verify-object* action requires computing $P(\text{obj}(B)$ that bounding box B is tight around *some* object. Models such as [1] can be used.

(6) New object. Finally, transition probabilities for a *name-box* action requires $P(\text{new}(B,\mathcal{C})|I)$ for a bounding box B as the probability that there is an object in this box which has not yet been named in the current set of classes $\mathcal{C}$. Assuming independence (and implicitly conditioning on I):

$$P(\text{new}(B,\mathcal{C})) = P(\text{obj}(B)) \prod_{C\in\mathcal{C}} (1 - P(\text{det}(B,C))) \quad (9)$$

The supplementary material provides additional details. Adding new human tasks in Table 1 would likely require the addition of new computer vision models.

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 is median human time in seconds.

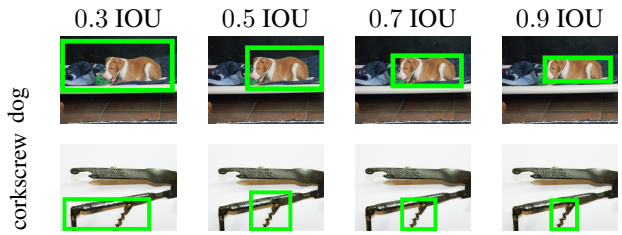

Figure 4. Bounding boxes with increasing intersection over union (IOU) with the optimal tight box. Training human annotators to make binary decision on whether or not a bounding box is a good detection is quite difficult; this the primary contributor to human error rates. Guidance such as "the object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

5. Experiments

We evaluate both the accuracy and cost of our proposed object annotation system that combines 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 (Section 5.2), showcase the quality of annotation obtained by our system (Section 5.3) and conclude by proving that our system is capable of self-monitoring (Section 5.4).

5.1. Setup

We perform experiments on the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) detection dataset [43]. 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]. We use 2216 images of val2 that contain at least 4 ground truth object instances. The average number of instances per image is 7.0 compared to 7.7 of COCO [37], and the average object size is 9.8% of image area compared to 10.5% in SUN [64].

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 [27] on ILSVRC2013 detection training set (full images, no bounding box labels) [24]. Detections and classifications with probability less than 0.1 are discarded.[5] We use non-maximum suppression on the output of detectors to (1) avoid multiple detections around the same instance, and (2) reduce the computational burden. The probability distribution for $P(\text{more}|n \text{ inst})$ is computed empirically for all classes jointly on the val1 set. The probability $P(\text{morecls}|n \text{ classes})$ is from [22].

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. [43, 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.[6]

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 compute the expected output of an action (Section 4.2) we need to 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. We generated positive and negative sets of examples for each task from ILSVRC val1 annotations,[7] and showed them to AMT workers. Quality control was done with a few "gold

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

[6] 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. [43] For an 200x200 pixel object, this corresponds to approximately 0.90 intersection over union.

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

standard" 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 labeling quality

We evaluate our proposed annotation system in simulation using the human accuracy rates collected in Section 5.2 to simulate the real-world labeling scenario. Figure 5 shows the average number of objects labeled as a function of budget (human labeling time). For the purposes of simulation, since only the 200 object category names in the image are known, we omit the verify-object and name-object tasks. We reach several conclusions based on these results:

Computer vision and human input are mutually beneficial. The object detectors (*CV only* in Figure 5) are able to label on average 0.95 objects per image at zero cost. Human-only annotation (*H only*) starts at zero labeled objects but improves over time. After 30 seconds of annotation, our joint method (*CV+H*) labels 1.5x more objects than the computer vision-only method and 2.8x more objects than the human-only method (Figure 5 left). This means that given 30 seconds of human time per image, adding in computer vision input can almost triple the accuracy of the human labeling.[8]

An MDP is an effective model for selecting human tasks. Figure 5 (right) shows that selecting questions at random is a surprisingly effective strategy that can label 3.9 ± 0.4 objects on average after 600 seconds of labeling (*CV+H: rand*). Our MDP approach significantly outperforms this baseline, labeling 6.0 ± 0.3 objects after 600 seconds.

Complex human tasks are necessary for effective annotation. We demonstrate that simple binary tasks are ineffective in our setting, by considering an MDP with just the verify-image and verify-box tasks (*CV+H: vi,vb* in Figure 5 (right). It labels 1.5x more objects than the CV-only baseline after 4.5 minutes of labeling and then plateaus. Our full system with all human tasks (*CV+H: all tasks*) achieves this improvement after just 1 minute, and then further improve to label 6.3x more objects than CV-only.

[8] Given a large labeling budget (Figure 5 right), human feedback contributes more than computer vision to the labeling. In fact, the human-only method even slightly outperforms the joint method in this case, partially due to the fact that it's difficult to perfectly calibrate object detectors to estimate their level of uncertainty (this is where e.g., [65] may be useful).

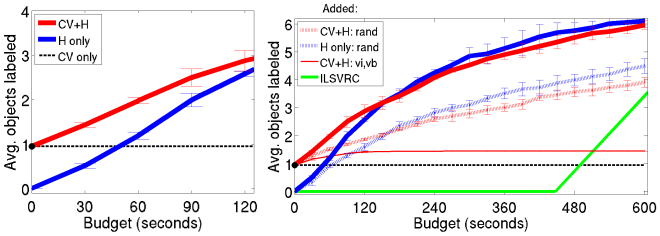

Figure 5. Our computer vision+human model (*CV+H*) compares favorably with others. The number of labeled objects (y-axis) is computed by averaging across multiple levels of precision on each image and then across all test images. Error bars correspond to one standard deviation across simulation runs. *Left.* The joint model handily outperforms the human-only (*H only*) and vision-only (*CV only*) baselines at low budget. *Right.* Our principled MDP is significantly better than choosing questions at random (*rand*). Variety of human interventions is critical; using only *verify-image* and *verify-box* human tasks is insufficient (*CV+H:vi,vb*). Our model also outperforms the ILSVRC-DET annotation baseline of [43].

Our annotation strategy is more effective than the ILSVRC-DET system [43]. The ILSVRC detection system consists of two steps: (1) determining what object classes are present in the images, followed by (2) asking users to draw bounding boxes. (1) is described in [13, 43]. Using their annotation times, hierarchical annotation method and optimistically assuming just 2.5 workers per label, determining the presence/absence of all 200 object categories with 95% accuracy would take 446.9 seconds per image. [50] describes Step 2 and reports time per bounding box (including quality control) as 42.4 seconds. This baseline is shown in green in Figure 5 (right), and labels 3.6 objects after 600 seconds, on par with our random baseline (*CV+H: rand*) and significantly below our joint model.[9]

Figure 6 shows qualitative results of our labeling system.

5.4. Satisfying 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 7(a) plots requested budget (x-axis) and requested precision (colored lines) versus the utility of returned label-

[9] One important difference to note is that the ILSVRC-DET system was optimized for annotating the desired 200 object classes while our system allows users to freely name new object classes.

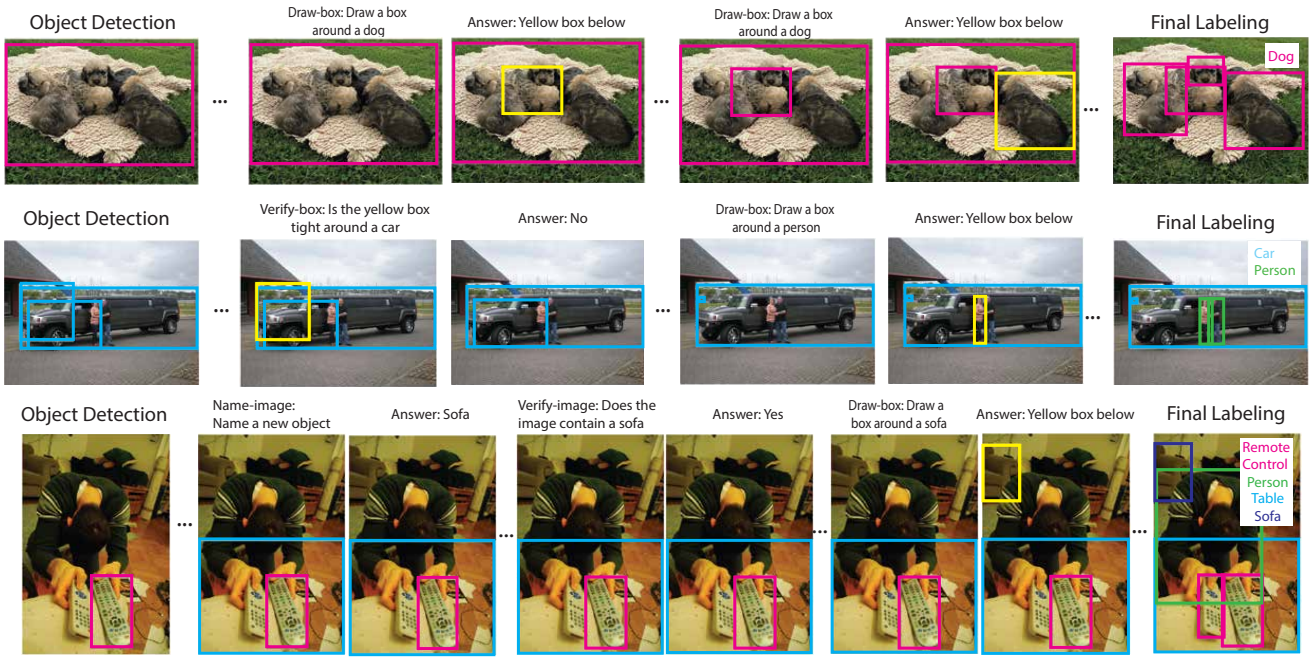

Figure 6. Some example results from our system integrating computer vision with multiple types of user feedback to annotate objects.

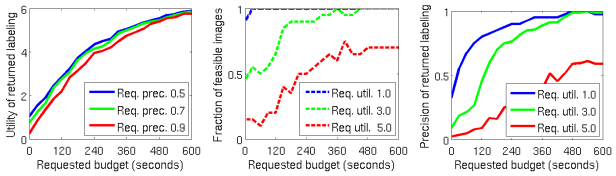

Figure 7. Quality of returned labeling as a function of requester constraints; details in Section 5.4.

ing. 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.

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 7(b). 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 7(c) 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**, current object detectors are far from perfect and can only detect a couple of objects in an average image. Further, accuracy drops rapidly when a tighter bounding box (with IOU higher than 0.5) is required. Our work can be used for collecting large-scale datasets with minimal supervision to improve the current state-of-the-art object detectors; in turn, the improved models will make our system more effective.

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 [43] 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 [37] dataset that asking slightly more complex human tasks (such as immediately asking users to put a dot on the object rather than merely asking if the object appears) 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.

Acknowledgements

We thank Lamberto Ballan, Jonathan Krause and Kevin Tang for great feedback. The UI design builds off the work of Justin Johnson. This research was partially supported by ONR MURI, by Yahoo! Labs and by NVIDIA (through donated GPUs).

References

[1] B. Alexe, T. Deselares, and V. Ferrari. Measuring the objectness of image windows. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2012. 1, 5

[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. 2

[3] A. Biswas and D. Parikh. Simultaneous active learning of classifiers & attributes via relative feedback. In *Computer Vision and Pattern Recognition (CVPR)*, 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, K. E. Hjorleifsson, and P. Perona. Active annotation translation. In *Computer Vision and Pattern Recognition (CVPR)*, 2014. 2

[6] 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 (ECCV)*, 2010. 1, 2, 4, 5

[7] Q. Chen, Z. Song, Z. Huang, Y. Hua, and S. Yan. Contextualizing object detection and classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 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, 3, 4

[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 *Computer Vision and Pattern Recognition (CVPR)*, 2013. 1, 2

[12] J. Deng, J. Krause, and L. Fei-Fei. Fine-grained crowdsourcing for fine-grained recognition. In *Computer Vision and Pattern Recognition (CVPR)*, 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 on Computer Vision*, 2011. 1

[15] S. Divvala, A. Farhadi, and C. Guestrin. Learning everything about anything: Webly-supervised visual concept learning. In *Computer Vision and Pattern Recognition (CVPR)*, 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. *International Journal on Computer Vision*, 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. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32, 2010. 5

[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 *Computer Vision and Pattern Recognition (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*, abs/1311.2524.v4, 2013. 1, 5

[21] B. Goodrich and I. Arel. Reinforcement learning based visual attention with application to face detection. In *Computer Vision and Pattern Recognition (CVPR)*, 2012. 3

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

[23] 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 *Workshop on Web-scale Vision and Social Media, European Conference on Computer Vision (ECCV)*, 2012. 2

[24] J. Hoffman, S. Guadarrama, E. Tzeng, J. Donahue, R. Girshick, T. Darrell, and K. Saenko. LSDA: Large scale detection through adaptation. *CoRR*, abs/1407.5035, 2014. 1, 2, 5, 6

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

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

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

[28] E. Kamar, S. Hacker, and E. Horvitz. Combining human and machine intelligence in large-scale crowdsourcing. In *AAMAS*, 2012. 2, 3

[29] S. Karayev, M. Fritz, and T. Darrell. Anytime recognition of objects and scenes. In *Computer Vision and Pattern Recognition (CVPR)*, 2014. 3

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

[31] M. Khodabandeh, A. Vahdat, G. Zhou, H. Hajimirsadeghi, M. J. Roshtkhari, G. Mori, and S. Se. Discovering human interactions in videos with limited data labeling. *CoRR*, abs/1502.03851, 2015. 2

[32] A. Kovashka, S. Vijayanarasimhan, and K. Grauman. Actively selecting annotations among objects and attributes. *International Conference on Computer Vision (ICCV)*, 2011. 2

[33] A. Krizhevsky, I. Sutskever, and G. Hinton. ImageNet classification with deep convolutional neural networks. In *NIPS*, 2012. 1, 5

[34] S. Lad and D. Parikh. Interactively guiding semi-supervised clustering via attribute-based explanations. In *European Conference on Computer Vision (ECCV)*. 2014. 1, 2

[35] C. Li, D. Parikh, and T. Chen. Automatic discovery of groups of objects for scene understanding. In *Computer Vision and Pattern Recognition (CVPR)*, 2012. 1

[36] L.-J. Li, R. Socher, and L. Fei-Fei. Towards total scene understanding:classification, annotation and segmentation in an automatic framework. In *Computer Vision and Pattern Recognition (CVPR)*, 2009. 1

[37] 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 *European Conference on Computer Vision (ECCV)*, 2014. 1, 2, 6, 8

[38] 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

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

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

[41] 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

[42] 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

[43] 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. *CoRR*, abs/1409.0575, 2014. 1, 2, 6, 7, 8

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

[45] 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

[46] 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, 3

[47] B. Siddiquie and A. Gupta. Beyond active noun tagging: Modeling contextual interactions for multi-class active learning. In *Computer Vision and Pattern Recognition (CVPR)*, 2010. 2

[48] Z. Song, Q. Chen, Z. Huang, Y. Hua, and S. Yan. Contextualizing object detection and classification. In *Computer Vision and Pattern Recognition (CVPR)*, 2011. 1

[49] M. Spain and P. Perona. Some objects are more equal than others: Measuring and predicting importance. In *European Conference on Computer Vision (ECCV)*, 2008. 2

[50] H. Su, J. Deng, and L. Fei-Fei. Crowdsourcing annotations for visual object detection. In *AAAI Human Computation Workshop*, 2012. 1, 2, 7

[51] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. *CoRR*, abs/1409.4842, 2014. 5

[52] K. Tang, R. Sukthankar, J. Yagnik, and L. Fei-Fei. Discriminative segment annotation in weakly labeled video. In *Computer Vision and Pattern Recognition (CVPR)*, 2013. 2

[53] K. E. A. van de Sande, C. G. M. Snoek, and A. W. M. Smeulders. Fisher and vlad with flair. In *Computer Vision and Pattern Recognition (CVPR)*, 2014. 1

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

[55] S. Vijayanarasimhan and K. Grauman. Predicting effort vs. informativeness for multi-label image annotations. In *Computer Vision and Pattern Recognition (CVPR)*, 2009. 2

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

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

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

[59] C. Wah, S. Branson, P. Perona, and S. Belongie. Multiclass recognition and part localization with humans in the loop. In *International Conference on Computer Vision (ICCV)*, 2011. 1, 2

[60] 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)*, 2014. 1, 2

[61] X. Wang, M. Yang, S. Zhu, and Y. Lin. Regionlets for generic object detection. In *International Conference on Computer Vision (ICCV)*, 2013. 1

[62] 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

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

[64] J. Xiao, J. Hays, K. Ehinger, A. Oliva, and A. Torralba. SUN database: Large-scale scene recognition from Abbey to Zoo. *Computer Vision and Pattern Recognition (CVPR)*, 2010. 1, 2, 6

[65] P. Zhang, J. Wang, A. Farhadi, M. Hebert, and D. Parikh. Predicting failures of vision systems. In *Computer Vision and Pattern Recognition (CVPR)*, 2014. 7