Learning Intelligent Dialogs for Bounding Box Annotation

Ksenia Konyushkova
CVLab, EPFL
ksenia.konyushkova@epfl.ch *

Jasper Uijlings
Google AI Perception
jrru@google.com

Christoph H. Lampert
IST Austria
chl@ist.ac.at

Vittorio Ferrari
Google AI Perception
vittoferrari@google.com

Abstract

We introduce Intelligent Annotation Dialogs for bounding box annotation. We train an agent to automatically choose a sequence of actions for a human annotator to produce a bounding box in a minimal amount of time. Specifically, we consider two actions: box verification [34], where the annotator verifies a box generated by an object detector, and manual box drawing. We explore two kinds of agents, one based on predicting the probability that a box will be positively verified, and the other based on reinforcement learning. We demonstrate that (1) our agents are able to learn efficient annotation strategies in several scenarios, automatically adapting to the image difficulty, the desired quality of the boxes, and the detector strength; (2) in all scenarios the resulting annotation dialogs speed up annotation compared to manual box drawing alone and box verification alone, while also outperforming any fixed combination of verification and drawing in most scenarios; (3) in a realistic scenario where the detector is iteratively re-trained, our agents evolve a series of strategies that reflect the shifting trade-off between verification and drawing as the detector grows stronger.

1. Introduction

Many recent advances in computer vision rely on supervised machine learning techniques that are known to crave for huge amounts of training data. Object detection is no exception as state-of-the-art methods require a large number of images with annotated bounding boxes around objects. However, drawing high quality bounding boxes is expensive: The official protocol used to annotate ILSVRC [38] takes about 30 seconds per box [43]. To reduce this cost, recent works explore cheaper forms of human supervision such as image-level labels [6, 21, 53], box verification series [34], point annotations [27, 33], and eye-tracking [31].

Among these forms, the recent work on box verification series [34] stands out as it demonstrated to deliver high quality detectors at low cost. The scheme starts from a given weak detector, typically trained on image labels only, and uses it to localize objects in the images. For each image, the annotator is asked to verify whether the box produced by the algorithm covers an object tightly enough. If not, the process iterates: the algorithm proposes another box and the annotator verifies it.

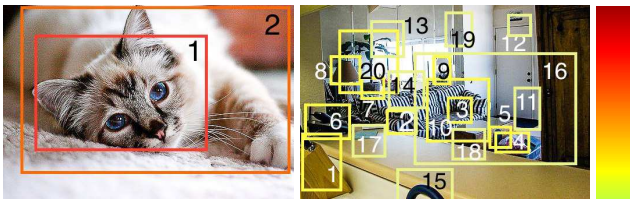

Figure 1: Left: an image with a target class *cat*. The weak detector identified two box proposals with high scores. The best strategy in this case is to do a series of box verifications. Right: an image with a target class *potted plant*. The weak detector identified many box proposals with low scores. The best strategy is to draw a box.

The success of box verification series depends on a variety of factors. For example, large objects on homogeneous backgrounds are likely to be found early in the series, and hence require little annotation time (Fig. 1, left). However, small objects in crowded scenes might require many iterations, or could even not be found at all (Fig. 1, right). Furthermore, the stronger the detector is, the more likely it is to correctly localize new objects, and to do so early in the series. Finally, the higher the desired box quality (i.e. how tight they should be), the lower the rate of positively verified boxes. This causes longer series, costing more annotation time. Therefore, in some situations manual box drawing [32, 43] is preferable. While more expensive than one verification, it always produces a box annotation. When an annotation episode consists of many verifications, its duration can be longer than the time to draw a box, depending on the relative costs of the two actions. Thus, different forms of annotation are more efficient in different situations.

In this paper we introduce Intelligent Annotation Dialogs (IAD) for bounding box annotation. Given an image, detector, and target class to be annotated, the aim of IAD is to automatically choose the sequence of annotation actions that

*This work was done during an internship at Google AI Perception

results in producing a bounding box in the least amount of time. We train an IAD agent to select the type of action based on previous experience in annotating images. Our method automatically adapts to the difficulty of the image, the strength of the detector, the desired quality of the boxes, and other factors. This is achieved by modeling the episode duration as a function of problem properties. We consider two alternative ways to do this, either a) by predicting whether a proposed box will be positively or negatively verified (Sec. 4.1), or b) by directly predicting the episode duration (Sec. 4.2).

We evaluate IAD by annotating bounding boxes in the PASCAL VOC 2007 dataset [15] in several scenarios: a) with various desired quality levels; b) with detectors of varying strength; and c) with two ways to draw bounding boxes, including a recent method which only takes 7s per box [32]. In all scenarios our experiments demonstrate that thanks to its adaptive behavior IAD speeds up box annotation compared to manual box drawing alone, or box verification series alone. Moreover, it outperforms any fixed combination of them in most scenarios. Finally, we demonstrate that IAD learns useful strategies in a complex realistic scenario where the detector is continuously improved with the growing amount of the training data. Our IAD code is made publicly available[1].

2. Related work

Drawing bounding boxes Fully supervised object detectors are trained on data with manually annotated bounding boxes, which is costly. The reference box drawing interface [43] used to annotate ILSVRC [38] requires 25.5s for drawing one box. Recently, a more efficient interface reduces costs to 7.0s without compromising on quality [32]. We consider both interfaces in this paper.

Training object detectors from image-level labels Weakly Supervised Object Localization (WSOL) [6, 13, 14, 16, 21, 53] methods train object detectors from images labeled only as containing certain object classes, but without bounding boxes. This avoids the cost of box annotation, but leads to considerably weaker detectors than their fully supervised counterparts [6, 16, 21, 53]. To produce better object detectors, extra human annotation is required.

Other forms of weak supervision Several works aim to reduce annotation cost of manual drawing by using approximate forms of annotation. Eye-tracking is used for training object detectors [31] and for action recognition in videos [26]. Point-clicks are used to derive bounding box annotations in images [33] and video [27], and to train semantic segmentation models [2, 3, 50]. Other works train semantic segmentation models using scribbles [25, 51].

In this paper we build on box verification series [34], where boxes are iteratively proposed by an object detector and verified by a human annotator. Experiments show that humans can perform box verification reliably (Fig. 6 of [34]). Besides, the Open Images dataset [23] contains 2.5 Million boxes annotated in this manner, demonstrating it can be done at scale.

Interactive annotation Several works use human-machine collaboration to efficiently produce annotations. These works address interactive segmentation [8, 37, 12, 18, 17, 30], attribute-based fine-grained image classification [10, 35, 7, 49], and interactive video annotation [48]. Branson *et al*. [9] transform different types of location information (e.g. parts, bounding boxes, segmentations) into each other with corrections from an annotator. These works follow a predefined annotation protocol, whereas we explore algorithms that can automatically select questions, adapting to the input image, the desired quality of the annotation, and other factors.

The closest work [39] to ours proposes human-machine collaboration for bounding box annotation. Given a repertoire of questions, the problem is modeled with a Markov decision process. Our work differs in several respects. (1) While Russakovsky *et al*. [39] optimizes the expected precision of annotations over the whole dataset, our method delivers quality guarantees on each individual box. (2) Our approach of Sec 4.1 is mediated by predicting the probability of a box to be accepted by an annotator. Based on this, we provide a provably optimal strategy which minimizes the expected annotation time. (3) Our reinforcement learning approach of Sec. 4.2 learns a direct mapping from from measurable properties to annotation time, while avoiding any explicit modelling of the task. (4) Finally, we address a scenario where the detector is iteratively updated (Sec. 5.3), as opposed to keeping it fixed.

Active learning (AL) In active learning the goal is to train a model while asking human annotations for unlabeled examples which are expected to improve the model accuracy the most. It is used in computer vision to train whole-image classifiers [20, 22], object class detectors [47, 52], and semantic segmentation [41, 45, 46]. While the goal of AL is to select a subset of the data to be annotated, this paper aims at minimizing the time to annotate each of the examples.

Reinforcement learning Reinforcement learning (RL) traditionally aims at learning policies that allow autonomous agents to act in interactive environments. Reinforcement learning has a long tradition e.g. in robotics [1, 28, 24]. In computer vision, it has mainly been used for active vision tasks [11, 4, 19], such as learning a policy for the spatial exploration of large images or panoramic images. Our use of RL differs from this, as we learn a policy for image annotation, not for image analysis. The learned policy enables the system to dynamically choose the annotation mechanism by which to interact with the user.

[1] https://github.com/google/intelligent_annotation_dialogs

3. Problem definition and motivation

3.1. Why use intelligent annotation dialogs?

In this paper we tackle the problem of producing bounding box annotations for a set of images with image-level labels indicating which object classes they contain. Consider annotating a *cat* in Fig. 1 (left). The figure shows two bounding boxes found by the detector. We notice that: a) the image is relatively simple with only one distinct object; b) there are only few high-scored *cat* detections; c) they are big; d) we might know a-priori that the detector is strong for the class *cat* and thus detections for it are often correct. As box verification is much faster than drawing, the most efficient way to annotate a box in this situation is with a box verification series.

Now consider instead annotating a *potted plant* in Fig. 1 (right). We notice that: a) the image is cluttered with many details; b) there are many low-scored *potted plant* detections; c) they are small; d) we might know a-priori that the detector is weak for this class and thus the detections for it are often wrong. In this situation, it is unlikely that the correct bounding box comes early in the series. Thus, manual box drawing is likely to be the fastest annotation strategy.

Even during annotation of one image-class pair, the best strategy may combine both annotation types: Given only one high-scored box for *cat*, the best expected strategy is to verify one box, and, if rejected, ask manual box drawing.

These examples illustrate that every image, class and detector output requires a separate treatment for designing the best annotation strategy. Thus, there is need for a method that can take advantage of this information to select the most time efficient sequence of annotation actions. In this paper, we propose two methods to achieve this with the help of Intelligent Annotation Dialog (IAD) agents. In our first approach (Sec. 4.1) we explicitly model the expected episode duration by taking into consideration the probability for each proposed box to be accepted. Our second approach (Sec. 4.2) casts the problem in terms of reinforcement learning and leans a strategy from trial-and-error interactions without an intermediate modeling step.

3.2. Problem definition

We are given an image with image-level labels that indicate which object classes it contains. We treat each class independently, and we want to produce one bounding box given a single image-class pair. In particular, given that the image contains a set $\mathcal{B}^\star$ of object instances of the target class, we want to produce a bounding box $\hat{b}$ of sufficient quality around one such object $b^\star$. We measure the quality in terms of Intersection-over-Union (IoU) and we want to find $\hat{b}$ such that there exists $b^\star \in \mathcal{B}^\star$: $\text{IoU}(\hat{b}, b^\star) \geq \alpha$. More specifically, we want to automatically construct a sequence of actions which produces $\hat{b}$ while minimizing annotation time,

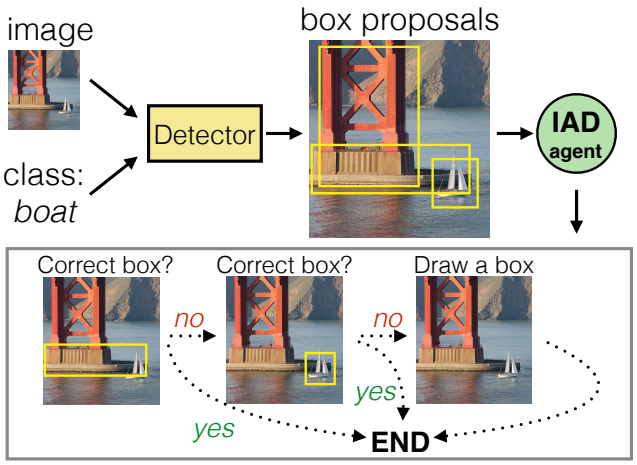

Figure 2: Intelligent Annotation Dialog agent in action. For a given image and class *boat* the detector identifies a set of box proposals. IAD agent produces a planned dialog V^2D that means that the first two box proposals are verified and if none of them is accepted, manual box drawing is done. In reality, the annotation terminates after two box verifications.

choosing from two annotation actions: manual bounding box drawing D [32, 43] that takes t_D seconds and bounding box verification V [34] that takes t_V seconds.

We design the annotation dialog to end with a successfully annotated bounding box. Logically, the only possible planned sequence of actions which does this has the form V^mD. No sequence of verification V is guaranteed to produce a bounding box, so if m verifications fail to produce one, manual drawing is required. Conversely, manual drawing always produces a box and the dialog ends. Fig 2 illustrates how IAD agent produces a planned sequence of V^2D for the task of detecting a *boat* in the image with several detections. In reality, only a sequence of actions V^2 is executed because a boat is found at the second verification.

Verification questions are generated using an object detector. Papadopoulos *et al*. [34] present the highest scored detection to the annotator. Upon rejection, they remove boxes which highly overlap with the rejected one (this procedure is called *search space reduction*), after which they present the next box with the highest score. In this paper we assume the detector stays constant during a single annotation dialog, which means we can do search space reduction by non-maximum suppression (NMS). Let us denote by B_0 the sequence of detections followed by NMS. Because of NMS, we can assume boxes in B_0 to be independent for verification. Let $\mathcal{S}$ be the set of all possible sequences of distinct elements in B_0. Now our goal is to plan a sequence of actions $\pi = V^mD$ on a sequence $S_m = (s_1, \ldots, s_m) \in \mathcal{S}$.

We can now formally define the optimization criterion for the IAD agent. Let $t(V^mD, S_m)$ be the duration of the episode when strategy V^mD is applied to a sequence S_m

and let us denote its expected duration as $T(V^m D, S_m)$. The task of IAD is to choose 1) the maximum number of verifications $m = k$ that will define a sequence of actions $V^k D$, and 2) a sequence of boxes $A_k = (a_1, \ldots, a_k) \in S$ such that the duration of the episode is minimized in expectation:

$$T(V^k D, A_k) \leq T(V^m D, S_m), \quad m \in \{0, \ldots, n\}, \forall S_m \in \mathcal{S}. \tag{1}$$

4. Methods

We now present our two methods to construct Interactive Annotation Dialogs (IAD).

4.1. IAD by predicting probability of acceptance

One way to minimize the expected duration of the episode is by estimating the probability that the proposed boxes will be accepted by the annotator. We can train a classifier g that will predict if the box $b_i \in B_0$ is going to be accepted or not as a function of various parameters of the state of the episode. By looking at the probability of acceptance $p(b_i)$ for every box, we can compute the expected duration of the episode $T(V^m D, S_m)$ for any $V^m D$ and S_m. Given this acceptance probability estimation, we show that there exists a simple decision rule that chooses m and S_m so as to minimize the expected episode duration.

Optimal strategy Suppose for now that we know the probabilities $p(b_i)$ for every box b_i to be accepted at a quality level α:

$$p(b_i) = \mathbb{P}[\max_{b^\star \in B^\star} \{\text{IoU}(b_i, b^\star)\} \geq \alpha]. \tag{2}$$

Later in this section we will explain how to estimate $p(b_i)$ in practice.

Imagine for a moment that we have only one box proposal b_1. In this case the only two possible sequences of actions are D and $V^1 D$. Let us compute the expected time until the end of the episode for both of them. The episode duration for strategy D is just the time required for manual drawing: $T(D) = t_D$.

For the second strategy $V^1 D$, the end of the episode is reached with probability $p(b_1)$ when a box proposal is accepted and with probability $q(b_1) = 1 - p(b_1)$ when manual drawing is done. Hence, the expected duration of the episode is

$$T(V^1 D, (b_1)) = t_V + q(b_1) t_D. \tag{3}$$

As we want to choose the strategy with the lowest expected duration of the episode, D is preferred to $V^1 D$ if $T(D) \leq T(V^1 D, (b_1))$, i.e.

$$t_D \leq t_V + q(b_1) t_D \iff p(b_1) \leq t_V / t_D. \tag{4}$$

Now let us go back to a situation with a sequence of box proposals B_0. We sort B_0 in the order of decreasing probability of acceptance $p(b_i)$, resulting in a sequence of boxes $\bar{S}_n$. Consider the following strategy (Alg. IAD-Prob): Verify boxes from $\bar{S}_n$ for which $p(b_i) > t_V / t_D$; if none of them is accepted, then do manual box drawing. We claim that the strategy produced by IAD-Prob is optimal, i.e. it minimizes the expected duration of the episode.

Algorithm IAD-Prob

Input: $B_0 = (b_1, \ldots, b_n); p(b_1), \ldots, p(b_n); t_V; t_D$
$\bar{S}_n = (\bar{s}_1, \bar{s}_2, \ldots, \bar{s}_n) \leftarrow \text{sort}(B_0)$ by $p(b_i)$
$\pi = ()$
$A_k = ()$
while $p(\bar{s}_i) > t_V / t_D$ **do**
  $A_k \leftarrow A_k \frown \bar{s}_i$
$\pi \leftarrow V^k D$
return sequence of actions π, sequence of boxes A_k

Theorem 1. *If probabilities of acceptance $\{p(b_i)\}$ are known, the strategy of applying a sequence of actions $V^k D$ defined by IAD-Prob to a sequence of boxes A_k minimizes the annotation time, i.e. for all $m \in \{0, \ldots, n\}$ and for all box sequences S_m:*

$$T(V^k D, A_k) \leq T(V^m D, S_m) \tag{5}$$

Sketch of the proof. The proof consists of two parts. First, we show that for any strategy $V^m D$, the best box sequence is obtained by sorting the available boxes by their probability of acceptance and using the first m of them. Second, we show that the number of verification steps found by IAD-Prob, k, is indeed the optimal one.

We start by rewriting the expected episode length in closed form. For a strategy $V^m D$ and any sequence of boxes, $S_m = (s_1, \ldots, s_m)$, we obtain

$$\begin{aligned} T(V^m D, S_m) &= t_V + q(s_1) t_V + q(s_1) q(s_2) t_V + \ldots \\ &\quad + q(s_1) q(s_2) \cdots q(s_{m-1}) t_V + q(s_1) q(s_2) \cdots q(s_m) t_D \\ &= t_V \sum_{l=0}^{m-1} \prod_{j=1}^{l} q(s_j) + t_D \prod_{j=1}^{m} q(s_j). \end{aligned} \tag{6}$$

Our first observation is that (6) is monotonically decreasing as a function of $q(s_1), \ldots, q(s_m)$. Consequently, the smallest value is obtained by selecting the set of m boxes that have the smallest rejection probabilities. To prove that their optimal order is sorted in decreasing order, assume that S_m is not sorted, i.e. there exists an index $l \in \{1, \ldots, m-1\}$ for which $q(s_l) > q(s_{l+1})$. We compare the expected episode length of S_m to that of a sequence $\tilde{S}_m$ in which s_l and s_{l+1} are at switched positions. Using (6) and noticing that many

of the terms cancel out, we obtain

$$T(V^m D, S_m) - T(V^m D, \tilde{S}_m) = t_V \big(q(s_l) - q(s_{l+1})\big)\Big(\prod_{j=1}^{l-1} q_j\Big) \quad > 0. \tag{7}$$

This shows that $\tilde{S}_m$ has strictly smaller expected episode length than S_m, so S_m cannot have been the optimal order.

Consequently, for any strategy $V^m D$, the optimal sequence is to sort the boxes by decreasing probability of rejection, *i.e*. increasing acceptance probability. We denote it by $\bar{S}_m = (\bar{s}_1, \ldots, \bar{s}_m)$.

Next, we show that the number, k, of verification actions found by the IAD-Prob algorithm is optimal, *i.e*. $V^k D$ is better or equal to $V^m D$ for any $m \neq k$. As we already know that the optimal box sequence for any strategy $V^m D$ is $\bar{S}_m$, it is enough to show that

$$T(V^{m-1} D, \bar{S}_{m-1}) \geq T(V^m D, \bar{S}_m), \tag{8}$$

for all $m \in \{1, \ldots, k\}$, and

$$T(V^{m-1} D, \bar{S}_{m-1}) \leq T(V^m D, \bar{S}_m). \tag{9}$$

for all $m \in \{k+1, \ldots, n\}$. To prove these inequalities, we again make use of expression (6). For any $m \in \{1, \ldots, n-1\}$ we obtain

$$\begin{aligned} &T(V^m D, \bar{S}_m) - T(V^{m-1} D, \bar{S}_{m-1}) \\ &= t_V \prod_{j=1}^{m-1} q(\bar{s}_j) + t_D \prod_{j=1}^{m} q(\bar{s}_j) - t_D \prod_{j=1}^{m-1} q(\bar{s}_j) \\ &= \Big(\prod_{j=1}^{m-1} q(\bar{s}_j)\Big)\big(t_V + q(\bar{s}_m) t_D - t_D\big). \end{aligned} \tag{10}$$

For $m \in \{1, \ldots, k\}$, we know that $p(\bar{s}_m) > t_V / t_D$ by construction of the strategy. As in (4), this is equivalent to $t_V + q(\bar{s}_m) t_D - t_D \geq 0$. Consequently, (10) is nonnegative in this case, and inequality (8) is confirmed. For $m \in \{k+1, \ldots, n\}$, we know $p(\bar{s}_m) \leq t_V / t_D$, again by construction. Consequently, $t_V + q(\bar{s}_m) t_D - t_D \leq 0$, which shows that (10) is nonpositive in this case, confirming (9). □

Predicting acceptance probability To follow the optimal strategy IAD-Prob, we need the probabilities of acceptance $\{p(b_i)\}$ which we estimate using a classifier g. To obtain these probabilities we start with a (small) set Z_0 of annotated bounding boxes on a set of images I_0. We apply a detector f_0 on I_0 to obtain a set of detections B_0. Afterwards, we generate a feature vector ϕ_i for every box $b_i \in B_0$. The exact features are specified in Sec. 5.1 and include measurements such as detector scores, entropy, and box-size.

Next, we simulate verification responses for box proposals B_0 of every image-class pair with known ground truth. A box b_i gets label $y_i = 1$ if its IoU with any of the ground truth boxes is great or equal to α, otherwise it gets label 0. This procedure results in feature-label pairs (ϕ_i, y_i) that serve as a dataset for training a probabilistic classifier g.

Intuitively, the classifier learns that, for example, boxes with high detector's score are more likely to be accepted than boxes with low detector's score, bounding boxes for class *cat* are more likely to be accepted than bounding boxes for class *potted plant*, and smaller bounding boxes are less likely to be accepted than big ones.

4.2. IAD by reinforcement learning

The problem of finding a sequence of actions to produce a box annotation can be naturally formulated as a reinforcement learning problem. This approach allows us to learn a strategy directly from trial-and-error experience and to avoid the explicit modeling of Sec. 4.1. To construct an optimal strategy it does not need any prior knowledge about the environment. Thus, it is easily extensible to other types of actions or to stochastic environments with variable response time by an annotator.

Suppose that bounding boxes in an episode are verified in order of decreasing detector's score given by B_0. In an *episode* of annotating one image for a given target class, the IAD agent interacts with the *environment* in the form of the annotator. A *state* s_τ is characterised by the properties of a current image, detector and a current box proposal (as ϕ_i in Sec. 4.1). In each state the agent has a choice of two possible *actions* a: 1) ask for verification of the current box ($a = V$) and 2) ask for a manual drawing ($a = D$) The *reward* at every step τ is the negative time required for the chosen action: $r_\tau = -t_V$ and $r_\tau = -t_D$. If a box is positively verified or manually drawn, the episode terminates with a reward 0. Otherwise the agent finds itself in the next state corresponding to the next highest-scored box proposal in B_0. The total *return* of the episode is the sum of rewards over all steps. Denoting the number of steps after which an episodes terminates by K, the return is $R = \sum_{\tau=1}^{K} r_\tau$. This is equal to $-(K-1)t_V - t_D$ if the episode finished with manual drawing, or $-K t_V$ if it finished with box acceptance. By trying to maximise the return R, the agent learns a policy π that minimises the total annotation time. This results in a strategy that consists of a sequence of actions π applied to a sequence of boxes B_0.

Training the agent The agent can learn the optimal policy π from trial and error interactions with the environment. As in Sec. 4.1, we train on a small subset of annotated bounding boxes Z_0. We learn a policy with *Q-learning* which learns to approximate *Q-function* $Q^\pi(a, s_\tau)$ that indicates what return the agent should expect at state s_τ after taking an action a and after that following a strategy π.

5. Experiments

5.1. Experimental setup

We evaluate the performance of the IAD approach on the task of annotating bounding boxes on the PASCAL VOC 2007 trainval dataset. In all experiments our detector is Faster-RCNN [36] using Inception-ResNet [44] as base network.

Annotator actions and timings We simulate the annotator based on the ground truth bounding boxes. When asked for verification, a simulated human annotator deterministically accepts a box proposal if IoU $\geq \alpha$ and it takes $t_V = 1.8$ seconds [34]. When the simulated annotator is asked to draw a box, we use the ground truth box. We consider two interfaces for drawing: the classical manual drawing M [43] and the new faster Extreme Clicking X [32]. We consider that it takes a simulated user $t_M = 25.5$ or $t_X = 7$ seconds to return a bounding box that corresponds to any of the objects $b^\star$ [43, 32].

Box proposal order The order of box proposals for verifications is set to be B_0, i.e. in decreasing order of detector's score (Sec. 3.2). Then, the optimality condition of strategy IAD-Prob assumes that a box with higher score is more likely to be accepted than a box with lower score. Empirically, we observe only rare cases when this assumption is violated, but even then, changing the order does not improve results. Thus, we keep the original order B_0 for computational efficiency and consistency with IAD-RL. The images come in the same fixed random order for all methods.

Box features When predicting the acceptance probability (Sec. 4.1) and during reinforcement learning (Sec. 4.2), we use the following features ϕ_i characterizing box b_i, image, detector, and target class: a) prediction score of the detector on the box: $d(b_i)$; b) relative size of the box b_i in the image; c) average prediction score of all box proposals for the target class; d) difference between c) and $d(b_i)$; e) difference between the maximum score for the target class among all box proposals and $d(b_i)$; f) one-hot encoding of class.

IAD-Prob To predict box acceptance probabilities, we use a neural network classifier with 2 to 5 layers containing 5 to 50 neurons in each layer for predicting the acceptance of a box and these parameters are chosen in cross-validation (Sec. 4.1). We experimented with other types of classifiers including logistic regression and random forest and did not find any significant difference in their performance.

IAD-RL We learn a policy for the reinforcement learning agent with a method similar to [29] (Sec. 4.2). The function approximation of Q-values is a fully-connected neural network with 2 layers and 30 neurons at every layer. We learn it from interactions with simulated environment using experience replay. We use exploration rate $\epsilon = 0.2$, mini-batches of size 64 and between 500 and 1000 training iterations. A subset of training samples is reserved for validation: we use it for choosing parameter of neural network and for early stopping.

5.2. IAD with a fixed detector

Scenarios We evaluate our methods in several scenarios, by varying the following properties of the problem: a) the desired quality of boxes, b) the strength of the detector, and c) which interface is used to draw a box. Intuitively, different properties tend to prioritize different actions V or D. The higher the desired quality is, the more frequently manual box drawing is needed. When the detector is strong, box verification is successful more often and is preferred to drawing due to its small cost. Finally, using the fast Extreme Clicking interface, manual drawing is cheaper and becomes more attractive. Specifically, we consider the following three configurations, each for both quality levels:

1. **Weak detector, slow drawing, varying quality**
   Classical, slow interface to draw boxes [43] with a weak detector. To train the detector (Sec. 5.1), we first produce bounding box estimates using standard Multiple Instance Learning (MIL, e.g. [5, 13, 42]). The first two columns of Tab. 1 report the average time per one annotation episode.

2. **Weak detector, fast drawing, varying quality**
   The fast Extreme Clicking [32] for drawing boxes. We report the results in columns 3 and 4 of Tab. 1.

3. **Strong detector, fast drawing, varying quality**
   In many situations we have access to a reasonably strong detector before starting annotation of a new dataset. To model this we train f_0 on the PASCAL 2012 dataset train set which contains 16k boxes. The results are presented in the last two columns of Tab. 1.

Dataset We use PASCAL 2007 trainval [15], where we assume that image-level annotations are available for all images, whereas bounding boxes are given only in a small subset of images Z_0. The task is to annotate the rest of the images Z' with bounding boxes. Z and Z' are set with 10-fold validation and the reported results are averages over them.

Standard strategies As baselines, we consider two standard annotation strategies. The first is to always do manual drawing (D). The second is to run box verification series, followed by drawing if all available boxes have been rejected (V^*D). This strategy is guaranteed to terminate successfully while being the closest to [34].

Fixed strategies We introduce a family of fixed strategies that combine the two actions V and D in a predefined manner, without adapting to a particular image, class and detector: V^1D, V^2D, and V^3D.

Drawing technique	Slow drawing		Fast drawing			
Detector	Weak detector		Weak detector		Strong detector	
Quality level	$\alpha = 0.7$	$\alpha = 0.5$	$\alpha = 0.7$	$\alpha = 0.5$	$\alpha = 0.7$	$\alpha = 0.5$
D (standard)	25.50 ± 0.00	25.50 ± 0.00	$\mathbf{7.00} \pm 0.00$	7.00 ± 0.00	7.00 ± 0.00	7.00 ± 0.00
V^1D	$\mathbf{23.01 \pm 0.07}$	17.30 ± 0.07	7.62 ± 0.02	$\mathbf{6.05} \pm 0.02$	$\mathbf{3.45 \pm 0.01}$	2.50 ± 0.01
V^2D	23.79 ± 0.06	16.67 ± 0.06	8.92 ± 0.02	6.67 ± 0.02	3.48 ± 0.01	$\mathbf{2.45} \pm 0.01$
V^3D	24.67 ± 0.07	$\mathbf{16.38} \pm 0.07$	10.21 ± 0.02	7.32 ± 0.03	3.65 ± 0.02	2.48 ± 0.01
$V^\star D$ (standard)	42.29 ± 0.07	17.37 ± 0.07	31.82 ± 0.11	11.46 ± 0.04	8.83 ± 0.09	3.18 ± 0.02
IAD-Prob	23.07 ± 0.23	12.64 ± 1.29	6.81 ± 0.02	5.86 ± 0.04	3.42 ± 0.18	2.73 ± 0.08
IAD-RL	23.62 ± 0.38	16.30 ± 0.09	6.83 ± 0.03	5.89 ± 0.05	3.60 ± 0.07	2.66 ± 0.06
lower bound	18.55 ± 0.05	10.23 ± 0.04	5.99 ± 0.01	4.66 ± 0.01	2.80 ± 0.01	2.19 ± 0.01

Table 1: Average episode duration for *standard*, *fixed* and IAD strategies in scenarios varying in drawing speed, strength of detector and quality level. **Best fixed strategy** results are highlighted in bold. The best result of each scenario is indicated in yellow (multiple highlights if very close). The two IAD agents do approximately equally well.

Lower bound We also report the lower bound on the duration of the annotation episode. If we knew which box (Sec 3.2) in the proposal sequence B_0 is the first that will be accepted, we could choose a sequence of actions that leads to the lowest annotation cost. If accepted box is at the position $k^\star$ in sequence B_0, then the strategy is the following. If the cost of $k^\star$ verifications is lower than the cost of a drawing, then the verification series is done, otherwise, drawing is done. Note how this lower bound requires knowing the ground-truth bounding box. So, it is only intended to reveal the limits of what can be achieved by the type of strategies that we explore.

Results Tab. 1 shows that the scenario settings indeed influence the choice between V and D, along three dimensions: a) When annotations of higher quality are required, the best fixed strategy does fewer verifications, i.e. it resorts to manual drawing ealier in the series than when lower quality is acceptable (columns 1 vs. 2, 3 vs. 4, 5 vs. 6). b) When the detector is strong (columns 5 and 6), the best fixed strategy does more box verifications than with a weak detector (columns 3 and 4). c) When manual drawing is fast (columns 3 and 4), the best fixed strategy tends to do fewer box verifications than when drawing is slow (columns 1 and 2). The gap to the lower bound indicates how hard each of the scenarios is.

Importantly, both of our IAD strategies outperform any standard strategy in all scenarios. Moreover, IAD-Prob is significantly better than the best fixed strategy in three scenarios, equal in two, and worse in one. No single fixed strategy works well in *all* scenarios, and finding the best fixed strategy requires manual experimentation. In contrast, IAD offers a principled way to automatically construct an adaptive strategy that works well in *all* problem settings. Indeed, the consistent competitive performance of IAD demonstrates that it learns to adapt to the scenario at hand.

5.3. IAD with an iteratively improving detector

In realistic settings, the detector becomes stronger with a growing amount of annotations. Thus, to annotate bounding boxes with minimal cost, the object detector should be iteratively re-trained on previously annotated data.

Horizontal re-training One way to introduce detector re-training is suggested by the box verification series technique [34]. It starts with a given object detector f_0, typically trained on image-level labels using MIL. In the first iteration, f_0 is applied to all images, and the highest scored detection b_1 in each image is sent for human verification. After this, the detector is re-trained on all accepted boxes, giving a new detector f_1. In the second iteration, f_1 is applied to all images where a proposed box was rejected, attempting to localize the objects again as f_1 is stronger than f_0 (re-localization phase). Afterwards, these new detections are sent for verification, and finally the detector is re-trained again. The re-training, re-localization, and verification phases are iteratively alternated for a predefined number of iterations. We refer to this method as V*-hor* in our experiments. It essentially corresponds to the original method [34].

Vertical re-training A different way to incorporate detector re-training is inspired by batch-mode active learning [40]. In this case, a subset of images I_1 (batch) is annotated until completion, by running box verification series in each image while keeping the initial detector f_0 fixed. After this, the detector is re-trained on all boxes produced so far, giving f_1, and is then applied to the next batch I_2 to generate box proposals. The process iteratively moves from batch to batch until all images are processed.

IAD with vertical re-training It is straightforward to apply vertical retraining to any fixed dialog strategy. However, re-training the detector on more data increases the advantage of V over D, so a truly adaptive strategy should change as

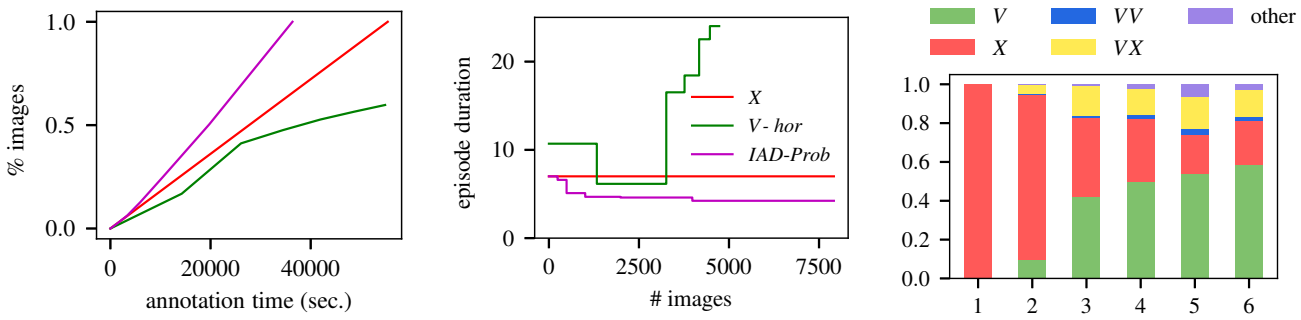

Figure 3: Left: the proportion of annotated images as a function of annotation time for IAD-Prob and *standard* strategies. Middle: average episode duration for various batches of data. Right: the proportion of various annotation sequences in batches of data at 6 iteration.

the detector gets stronger. We achieve this with the following procedure. At any given iteration τ, we train dialog strategy IAD-Prob$(I_\tau, f_{\tau-1})$ using boxes collected on I_τ and detector $f_{\tau-1}$. IAD-Prob$(I_\tau, f_{\tau-1})$ is applied with detector f_τ to collect new boxes on the next batch $I_{\tau+1}$. Note that IAD-Prob$(I_\tau, f_{\tau-1})$ is trained with the help of detector $f_{\tau-1}$, but it is applied with the box proposals of detector f_τ. This procedure introduces a small discrepancy, but it is not important when detectors f_τ and $f_{\tau-1}$ are sufficiently similar, which is the case in the experiments below. To initialize the procedure we set f_0 to be a weakly supervised MIL detector and we annotate I_1 by manual box drawing D.

We set the desired quality of bounding boxes to high (i.e. $\alpha = 0.7$) and we use Extreme Clicking for manual drawing. We perform 6 re-training iterations with an increasingly large batch size: $|I_1| = 3.125\%$, $|I_2| = 3.125\%$, $|I_3| = 6.25\%$, $|I_4| = 12.5\%$, $|I_5| = 25\%$, $|I_6| = 50\%$. This batching schedule is motivated by the fact that the gain in detector's performance after re-training is more noticeable when the previous training set is considerably smaller.

Results Fig. 3 (left) shows what proportion of boxes is collected as a function of total annotation time. We compare IAD-Prob against the strategy V*-hor* [34], and the standard fast drawing strategy X. IAD-Prob is able to annotate the whole dataset faster than any of the considered strategies. Fig. 3 (middle) shows the average episode duration in each batch. By design, the annotation time for strategy X is constant. For V*-hor*, after the first re-training iteration (from a weakly supervised to supervised detector) the average annotation cost grows because only difficult images are left to be annotated. On the contrary, annotation time for IAD decreases with every new batch because dialogs become stronger and box verifications become more successful.

Quality of boxes and resulting detector The data for training a detector and strategy in IAD includes both manually drawn boxes and boxes verified at IoU ≥ 0.7. More precisely, IAD data collection results in 44% drawn boxes and 56% verified boxes. The quality of the verified boxes reaches 83% mIoU. The detector trained on the boxes produced by IAD reaches 98% of the mAP of the detector trained on ground-truth boxes.

Evolution of adaptive strategies To gain better understanding of adaptive behaviour of IAD, we study the composition of sequences of actions produced during labelling of each batch. Fig. 3 (right) shows the proportion of images that are labelled by X, V, VX, VV and others sequences of actions. At the beginning of the process (batch 2), the vast majority of boxes is produced simply by asking for Extreme Clicking (X). It means that IAD learns that this is the best thing to do when the detector is weak. As the process continues, the detector gets stronger and IAD selects more frequently series composed purely of box verifications (V,VV), and mixed series with both actions (VX). Examples of the annotation dialogs are presented in the supplementary materials. This experiment demonstrates that IAD is capable of producing strategies that dynamically adapt to the change in problem property caused by the gradually improving detector. One cannot achieve this with any fixed strategy.

6. Conclusion

In this paper we introduced Intelligent Annotation Dialogs for the task of bounding box annotation. IAD automatically chooses a sequence of actions V^kD that results in time-efficient annotations. We presented two methods to achieve this. The first method models the annotation time by predicting the acceptance probability for every box proposal. The second method skips the modelling step and learns an efficient strategy directly from trial-and-error interactions. In the extensive experimental evaluation IAD demonstrated competitive performance against various baselines and the ability to adapt to multiple problem properties. In future work we would like to model variable annotation time and context switches.

References

[1] M. Asada, S. Noda, S. Tawaratsumida, and K. Hosoda. Purposive behavior acquisition for a real robot by vision-based reinforcement learning. *Machine Learning*, 23(2):279–303, 1996. 2

[2] A. Bearman, O. Russakovsky, V. Ferrari, and L. Fei-Fei. What's the point: Semantic segmentation with point supervision. In *ECCV*, 2016. 2

[3] S. Bell, P. Upchurch, N. Snavely, and K. Bala. Material recognition in the wild with the materials in context database. In *CVPR*, 2015. 2

[4] M. Bellver, X. G. i Nieto, F. Marques, and J. Torres. Hierarchical object detection with deep reinforcement learning. In *NIPS Workshop on Deep Reinforcement Learning*, 2016. 2

[5] H. Bilen, M. Pedersoli, and T. Tuytelaars. Weakly supervised object detection with posterior regularization. In *BMVC*, 2014. 6

[6] H. Bilen and A. Vedaldi. Weakly supervised deep detection networks. In *CVPR*, 2016. 1, 2

[7] A. Biswas and D. Parikh. Simultaneous active learning of classifiers & attributes via relative feedback. In *CVPR*, 2013. 2

[8] Y. Boykov and M. P. Jolly. Interactive graph cuts for optimal boundary and region segmentation of objects in N-D images. In *ICCV*, 2001. 2

[9] S. Branson, K. Hjörleifsson, and P. Perona. Active annotation translation. In *CVPR*, 2014. 2

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

[11] J. C. Caicedo and S. Lazebnik. Active object localization with deep reinforcement learning. In *ICCV*, pages 2488–2496, 2015. 2

[12] L. Castrejon, K. Kundu, R. Urtasun, and S. Fidler. Annotating object instances with a polygon-rnn. 2017. 2

[13] R. Cinbis, J. Verbeek, and C. Schmid. Multi-fold mil training for weakly supervised object localization. In *CVPR*, 2014. 2, 6

[14] T. Deselaers, B. Alexe, and V. Ferrari. Localizing objects while learning their appearance. In *ECCV*, 2010. 2

[15] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes (VOC) Challenge. *IJCV*, 2010. 2, 6

[16] M. Haußmann, F. Hamprecht, and M. Kandemir. Variational bayesian multiple instance learning with gaussian processes. In *CVPR*, 2017. 2

[17] S. Jain and K. Grauman. Click carving: Segmenting objects in video with point clicks. In *Proceedings of the Fourth AAAI Conference on Human Computation and Crowdsourcing*, 2016. 2

[18] S. D. Jain and K. Grauman. Predicting sufficient annotation strength for interactive foreground segmentation. In *ICCV*, 2013. 2

[19] D. Jayaraman and K. Grauman. Learning to look around. *arXiv preprint arXiv:1709.00507*, 2017. 2

[20] A. J. Joshi, F. Porikli, and N. Papanikolopoulos. Multi-class active learning for image classification. In *CVPR*, 2009. 2

[21] V. Kantorov, M. Oquab, M. Cho, and I. Laptev. Contextlocnet: Context-aware deep network models for weakly supervised localization. In *ECCV*, 2016. 1, 2

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

[23] I. Krasin, T. Duerig, N. Alldrin, V. Ferrari, S. Abu-El-Haija, A. Kuznetsova, H. Rom, J. Uijlings, S. Popov, A. Veit, S. Belongie, V. Gomes, A. Gupta, C. Sun, G. Chechik, D. Cai, Z. Feng, D. Narayanan, and K. Murphy. Openimages: A public dataset for large-scale multi-label and multi-class image classification. *Dataset available from* https://github.com/openimages/dataset, 2017. 2

[24] S. Levine, C. Finn, T. Darrell, and P. Abbeel. End-to-end training of deep visuomotor policies. *JMLR*, 17:1–40, 2016. 2

[25] D. Lin, J. Dai, J. Jia, K. He, and J. Sun. ScribbleSup: Scribble-supervised convolutional networks for semantic segmentation. In *CVPR*, 2016. 2

[26] S. Mathe and C. Sminchisescu. Dynamic eye movement datasets and learnt saliency models for visual action recognition. In *ECCV*, 2012. 2

[27] P. Mettes, J. C. van Gemert, and C. G. Snoek. Spot on: Action localization from pointly-supervised proposals. In *ECCV*, 2016. 1, 2

[28] J. Michels, A. Saxena, and A. Ng. High speed obstacle avoidance using monocular vision and reinforcement learning. In *ICML*, 2005. 2

[29] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller. Playing atari with deep reinforcement learning. In *NIPS Deep Learning Workshop*. 2013. 6

[30] N. S. Nagaraja, F. R. Schmidt, and T. Brox. Video segmentation with just a few strokes. In *ICCV*, 2015. 2

[31] D. P. Papadopoulos, A. D. F. Clarke, F. Keller, and V. Ferrari. Training object class detectors from eye tracking data. In *ECCV*, 2014. 1, 2

[32] D. P. Papadopoulos, J. R. Uijlings, F. Keller, and V. Ferrari. Extreme clicking for efficient object annotation. In *ICCV*, 2017. 1, 2, 3, 6

[33] D. P. Papadopoulos, J. R. Uijlings, F. Keller, and V. Ferrari. Training object class detectors with click supervision. In *CVPR*, 2017. 1, 2

[34] D. P. Papadopoulos, J. R. R. Uijlings, F. Keller, and V. Ferrari. We don't need no bounding-boxes: Training object class detectors using only human verification. In *CVPR*, 2016. 1, 2, 3, 6, 7, 8

[35] A. Parkash and D. Parikh. Attributes for classifier feedback. In *ECCV*, 2012. 2

[36] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *NIPS*, 2015. 6

[37] C. Rother, V. Kolmogorov, and A. Blake. Grabcut: interactive foreground extraction using iterated graph cuts. *SIGGRAPH*, 23(3):309–314, 2004. 2

[38] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. Berg, and L. Fei-Fei. ImageNet large scale visual recognition challenge. *IJCV*, 2015. 1, 2

[39] O. Russakovsky, L.-J. Li, and L. Fei-Fei. Best of both worlds: human-machine collaboration for object annotation. In *CVPR*, 2015. 2

[40] B. Settles. Active learning literature survey. *University of Wisconsin, Madison*, 2010. 7

[41] B. Siddiquie and A. Gupta. Beyond active noun tagging: Modeling contextual interactions for multi-class active learning. In *CVPR*, 2010. 2

[42] P. Siva and T. Xiang. Weakly supervised object detector learning with model drift detection. In *ICCV*, 2011. 6

[43] H. Su, J. Deng, and L. Fei-Fei. Crowdsourcing annotations for visual object detection. In *AAAI Human Computation Workshop*, 2012. 1, 2, 3, 6

[44] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. In *CVPR*, 2016. 6

[45] S. Vijayanarasimhan and K. Grauman. Multi-level active prediction of useful image annotations for recognition. In *NIPS*, 2008. 2

[46] S. Vijayanarasimhan and K. Grauman. What's it going to cost you?: Predicting effort vs. informativeness for multi-label image annotations. In *CVPR*, 2009. 2

[47] S. Vijayanarasimhan and K. Grauman. Large-scale live active learning: Training object detectors with crawled data and crowds. *IJCV*, 108(1-2):97–114, 2014. 2

[48] C. Vondrick, D. Patterson, and D. Ramanan. Efficiently scaling up crowdsourced video annotation. *IJCV*, 2013. 2

[49] C. Wah, G. Van Horn, S. Branson, S. Maji, P. Perona, and S. Belongie. Similarity comparisons for interactive fine-grained categorization. In *CVPR*, 2014. 2

[50] T. Wang, B. Han, and J. Collomosse. Touchcut: Fast image and video segmentation using single-touch interaction. *CVIU*, 2014. 2

[51] J. Xu, A. G. Schwing, and R. Urtasun. Learning to segment under various forms of weak supervision. In *CVPR*, 2015. 2

[52] A. Yao, J. Gall, C. Leistner, and L. Van Gool. Interactive object detection. In *CVPR*, 2012. 2

[53] Y. Zhu, Y. Zhou, Q. Ye, Q. Qiu, and X. Jiao. Soft proposal networks for weakly supervised object localization. In *ICCV*, 2017. 1, 2