Toward Interactive Self-Annotation For Video Object Bounding Box: Recurrent Self-Learning And Hierarchical Annotation Based Framework

Trung-Nghia Le *[1], Sugimoto Akihiro[2], Shintaro Ono[1], and Hiroshi Kawasaki[3]

[1]University of Tokyo, Japan
[2]National Institute of Informatics, Japan
[3]Kyushu University, Japan

Abstract

Amount and variety of training data drastically affect the performance of CNNs. Thus, annotation methods are becoming more and more critical to collect data efficiently. In this paper, we propose a simple yet efficient Interactive Self-Annotation framework to cut down both time and human labor cost for video object bounding box annotation. Our method is based on recurrent self-supervised learning and consists of two processes: automatic process and interactive process, where the automatic process aims to build a supported detector to speed up the interactive process. In the Automatic Recurrent Annotation, we let an off-the-shelf detector watch unlabeled videos repeatedly to reinforce itself automatically. At each iteration, we utilize the trained model from the previous iteration to generate better pseudo ground-truth bounding boxes than those at the previous iteration, recurrently improving self-supervised training the detector. In the Interactive Recurrent Annotation, we tackle the human-in-the-loop annotation scenario where the detector receives feedback from the human annotator. To this end, we propose a novel Hierarchical Correction module, where the annotated frame-distance binarizedly decreases at each time step, to utilize the strength of CNN for neighbor frames. Experimental results on various video datasets demonstrate the advantages of the proposed framework in generating high-quality annotations while reducing annotation time and human labor costs.

1. Introduction

Deep learning methods require a large amount of training data with ground truth, although developments of deep learning bring benefits to a wide range of practical applications such as autonomous vehicle [10], anomaly detection [25], object tracking [44], object detection [39], scene understanding [20], and trajectory prediction [32]. The reason is that the amount and variety of training data drastically affect the performance of convolutional neural networks (CNNs). In the domain of autonomous driving and intelligent transportation systems, localizing all moving vehicles and persons on street scenes is crucial. To achieve this goal, providing ground-truth bounding boxes is essential for training and evaluating the performance of CNNs. Especially, video-related tasks [25, 32, 39] require a huge number of object annotations, namely, ground-truth bounding boxes of objects.

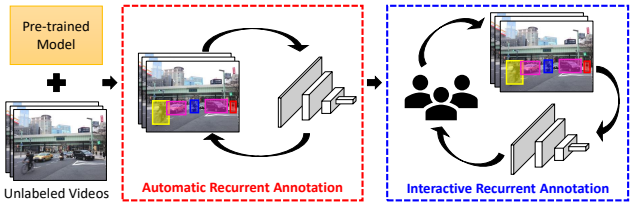

Figure 1. Our self-supervised learning based Interactive Self-Annotation framework for video object bounding box annotation consists of Automatic Recurrent Annotation, the supported process, and Interactive Recurrent Annotation, the main process.

Manually collecting object annotations is a time-consuming task. This becomes tedious when the target size is small, or the target is partly occluded in crowed scenes, which usually happens on street scenes. Indeed, drawing high-quality bounding boxes is extremely time-consuming, which typically requires annotators to spend around 50-80 seconds for each object [33] using Amazon Mechanical Turk (AMT) platform. Hence, it is essential for the development of effective annotation frameworks to generate desired ground-truth bounding boxes for large-scale video datasets.

In this paper, we propose a simple yet efficient In-

*Corresponding author. Email: ltnghia@its.iis.u-tokyo.ac.jp

teractive Self-Annotation (ISA) framework based on self-supervised learning to generate ground-truth bounding boxes for video objects. Our method can cut down both annotation time and human labor costs. The generated ground-truth information can be used for various tasks related to video objects. Our ISA framework consists of two recurrent annotation processes, i.e., Automatic Recurrent Annotation (ARA) and Interactive Recurrent Annotation (IRA), where ARA aims to build a supported detector for IRA. Each of these recurrent annotation processes aims to solve the data-detector problem in a learning loop: using the detector to update data and vice versa to improve both gradually. In the supported process ARA, we let an off-the-shelf detector watch unlabeled videos repeatedly to reinforce itself automatically. At each iteration, we introduce Labeling Assistant module to leverage both spatial information from a detector, which is trained from the previous iteration, and temporal consistency from tracking methods to select new pseudo ground-truth bounding boxes for self-supervised fine-tuning the detector. The new pseudo ground-truth bounding boxes have better quality than ones at the previous iteration, leading to improvement in training the detector. Meanwhile, the main process IRA tackles the human-in-the-loop annotation scenario, where feedback from the human annotator is utilized for training the detector, by incorporating human guidance into the automatic process. We propose a novel Hierarchical Correction module, where the annotated frame-distance binarizedly decreases at each time step, to utilize the strength of CNN for neighbor frames. The correction from the annotator is fed into training the detector for the next iteration to guide the detector back to the right track effectively. Extensive experiments on various video datasets confirm that our proposed framework has the ability to generate high-quality annotations while cutting down both annotation time and human labor costs.

The overall contribution of this paper is three-fold:

- We propose Interactive Recurrent Annotation (IRA) to allow a human annotator to easily interfere in the interaction-learning loop. Mistakes from a detector are corrected to guide the detector back to the right track at the next iteration. Our introduced Hierarchical Correction, in which the annotated frame-distance binarizedly decreases at each time step, is more efficient than the standard frame-by-frame annotation, resulting in the faster interactive annotation.
- We propose Automatic Recurrent Annotation (ARA) to support the main process IRA. In ARA, both spatial information from the trained detector from the previous iteration, and temporal consistency from tracking methods are leveraged to train the detector for the next iteration. Through the self-supervised learning on unlabeled videos, the detector can improve itself to generate higher quality of ground-truth bounding boxes gradually. The detector output from this process is used as an initial detector for the IRA process.
- We introduce a new metric, called Similarity in Union (SIU), to evaluate the task of object annotation. Our metric can be used to evaluate both bounding box and mask of objects.

The source code and annotated ground-truth of the datasets used in the experiments are publicly available on our project page. [1]

The remainder of this paper is organized as follows. In Section 2, we briefly review the related work. Next, our proposed framework is presented in Section 3. Experimental results are then reported and discussed in Section 4. Finally, Section 5 draws the conclusion and paves the way for future work.

2. Related Work

2.1. Video Object Annotation Tools

Considering the importance of ground-truth generation, we briefly review available video object annotation tools by the time order to show the evolution of the tools. ViPER (2000) [8] is an interface for manually annotating bounding boxes frame-by-frame, in which labels can be propagated in straightforward, consistent video frames, but it cannot deal with time-varying. LableME [28] is a popular web-based tool for annotating arbitrary shapes of an object, which has two versions: LableME-Image (2008) [28] for only image annotation, and LableME-Video (2009) [43] for video sequence annotation. VATIC (2013) [37], an online crowdsourcing video annotation tool of Amazon's Mechanical Turk, was developed to replace LableME-Video with better tracking and interpolation algorithms. Vondrick et al.[37] also designed a video annotation platform to annotate object bounding boxes using tracking. JAABA (2013) [15] is a semi-automated tool taking already annotated trajectories as input for labeling animal behaviors over video frames. iVAT (2015) [3], an interactive tool developed from VATIC, integrated computer vision algorithms working in an interactive and incremental learning framework to improve label propagation. ViTBAT (2016) [4] allows to annotates both individual-target and group-target objects. CVAT (2019) [29] has recently integrated supervised machine learning techniques to support interaction, and interpolation effectively for different tasks.

Although these tools support tracking algorithms, the human annotator is required to input the initial label of the target object manually, and then each object is tracked throughout its lifetime. On the other hand, our method can generate initial bounding boxes automatically, resulting in saving annotation time. CVAT (2019) [29] also can gener-

[1] https://sites.google.com/view/ltnghia/research/video_self_annotation

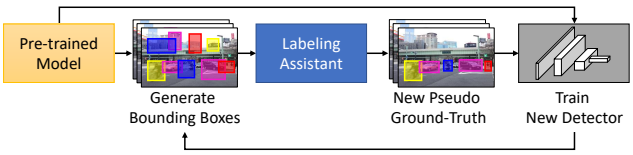

Figure 2. Pipeline of Automatic Recurrent Annotation based on self-supervised learning. The detector recurrently reinforce itself when watching unlabeled videos repeatedly.

ate initial labels automatically but from a fixed pre-trained model, thus it does not have the ability to adapt to new domains. Meanwhile, our proposed method learns contextual information from new unlabeled videos to adapt the detector to new domains. In addition, our method shows more advantages than existing tools through an interaction-learning loop. In this human-in-loop, our method suggests the human annotator correct key-frames and then learns these frames to better transfer labels to other video frames, resulting in cutting down annotation time.

2.2. Pseudo Data Based Learning

Amount and variety of training data drastically affect the performance of deep networks; thus, these methods always require a large number of training data. Due to the limited number of available training data, different methods have been proposed to generate pseudo data to train deep networks. Almost existing methods generate pseudo data from the real data and then combine all together to train networks. This approach is usually applied to solve one-shot learning problem in which semi-supervised video object segmentation methods [12, 34, 38] are trained from only the first video frame with given ground-truth. The standard method is to generate a large number of augmented data by transforming the labeled objects and then combining them with different background images [16, 35]. To forecast pedestrian trajectories, Olly et al. [32] jointly trained networks, using both human-annotated and machine-annotated bounding boxes, which generated by a pedestrian detection and tracking method. Miriam et al. [2] first trained an annotation network on real data to generate pseudo labels from unlabeled images and then combined both real and pseudo data to train the primary network for the task of semantic and instance segmentation. RoyChowdhury et al. [27] combined detection and tracking methods to generate pseudo labels for the target domain, and then combine them with existing labels from the source domain to adapt the trained detection network to new domains. On the other hand, Singh et al. [31] used motion cues to learn high precision object proposals and then trained a detection network on these proposals.

Mixing pseudo data with human-annotated data can help to train deep networks better thanks to the increasing number of training data, but this approach needs a certain number of given real data to generate pseudo data. In this paper, we solve this problem through a two-phase framework. In the first phase, we synthesize pseudo data through a self-supervised process by letting a detector watch a number of unlabeled videos repeatedly to reinforce itself automatically. In the second process, we incorporate human interaction to correct mistakes from the detector to guide the detector work on the right track recurrently.

3. Proposed Method

Figure 1 illustrates an overview of our proposed Interactive Self-Annotation (ISA) framework for video object bounding box annotation. The proposed framework consists of a supported process and a main process, namely Automatic Recurrent Annotation (ARA) and Interactive Recurrent Annotation (IRA), respectively. In the main process IRA, a human annotator interacts with a detector trained from the supported process ARA to speed up the interactive annotation.

3.1. Automatic Recurrent Annotation

3.1.1 Overview

Our goal is to generate bounding boxes for all objects in videos by letting a detector watch unlabeled videos repeatedly. At each iteration, both spatial information and temporal consistency in the videos are leveraged to train the detector at the next iteration. Through the self-supervised learning, the detector can improve itself automatically. This self-supervised learning can be applied to a group of videos, which have similar properties such as day/night, weather, and landscape, etc. In this way, the detector can learn shared information between similar videos to perform better. Our annotation method also can be used as an unsupervised detection method for domain adaptation.

Figure 2 illustrates the pipeline of our proposed Automatic Recurrent Annotation (ARA). At each iteration, the detector trained from the previous iteration generates bounding boxes for all videos. After that, we introduce the Labeling Assistant module to leverage temporal consistency from the videos to filter out these bounding boxes, resulting in new pseudo ground-truth bounding boxes for fine-tuning the detector again. The new pseudo ground-truth bounding boxes have better quality than ones at the previous iteration, which leads to improvement in training the detector for the next iteration.

3.1.2 Labeling Assistant

We aim to construct pseudo ground-truth bounding boxes to train a new detector at each iteration. We adopt the idea of RoyChowdhury et al. [27] that utilizes tracking algorithms to reinforce detection results with modification and

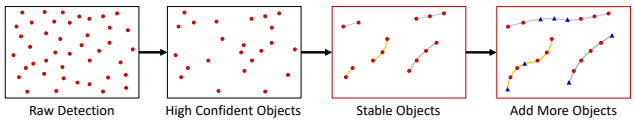

Figure 3. Pipeline of our Labeling Assistant module. Each dot is a detection result in the entire video. We first eliminate objects with low accuracy score and keep only high confident objects. After that, tubelets are constructed to remove all unstable objects with short temporal length. Remained objects in a tubelet are then assigned a unique label. Finally, we recover accidentally deleted objects as well as add miss-detected objects by the detector.

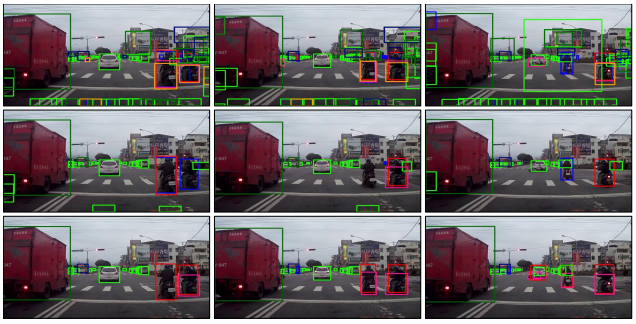

Figure 4. Visualization of Labeling Assistant module on a video sequence. From top to bottom, bounding boxes generated from the detector, high confident bounding boxes after thresholding, and pseudo ground-truth computed from our Labeling Assistant in this order. Labeling Assistant can remove noise detection, recover missing detection, and correct wrong labels by utilizing temporal motion in the video sequence.

improvement. We extend their method (Section 3.1 of [27]), which is proposed for a single category, to work with multiple categories. Figure 3 show our proposed Labeling Assistant module, which can select good bounding boxes from results generated by the trained detector from the previous iteration.

We first apply a confident threshold τ_s to eliminate objects with low accuracy score and *keep only high confident objects*. After that, frame-wise detections are associated across frames using a real-time tracking-by-detection method, namely DeepSort tracker [40], to create a series of bounding boxes with tracking identifies, namely tubelets, for each video. Objects, which are too small to track, are eliminated. We then *remove all unstable tubelets*, which have temporal length less than τ_t frames. We note that tracked objects in a tubelet can be at consistent frames or inconsistent frames. *Remained objects in a tubelet are assigned a unique label*, which has the largest total confident scores:

$$\hat{l} = \arg\max_{l \in L} \sum_{i \in tubelet} w_i^{(l)}, \tag{1}$$

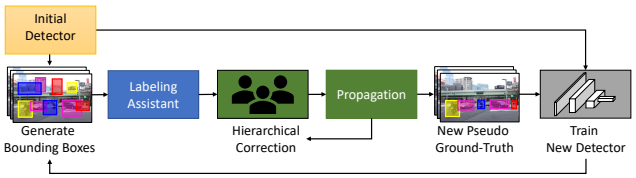

Figure 5. Pipeline of Interactive Recurrent Annotation process.

where L is the list of labels, and w denotes the detection accuracy of each object in the tubelet. Through this process, we can remove a large number of noise in the generated bounding boxes.

However, detected objects can be eliminated mistakenly during noise elimination. We aim to *recover accidentally deleted objects as well as add miss-detected objects by the detector* by expanding the set of detected objects of each tubelet in two directions, i.e., the next frames and the previous frames. Particularly, for each tubelet, we track the objects at the border of inconsistent frames towards the corresponding direction, using the SiamDW tracker [44], which can run in real-time. The tracker stops when IOU of the tracked box and ones at the neighbor frame less than the threshold τ_{iou}. Figure 4 illustrates examples of our proposed Labeling Assistant.

3.2. Interactive Recurrent Annotation

3.2.1 Overview

We aim to address the human-in-the-loop annotation scenario where the detector receives guidance from the human annotator to run on the right track. In this paper, we propose an Interactive Recurrent Annotation (IRA) (cf. Fig. 5), which leverages strengths of both ARA process and human annotator's guidance, resulting in low annotation cost. The main advantage of our method is that it allows a human annotator to easily interfere at any time if a mistake occurs. In particular, at each iteration, the human annotator has the possibility to correct wrong or missed detections, producing as accurate bounding boxes as desired by the human annotator. The correction is then fed back to the detector through self-supervised training, effectively helping the detector to get back to the right track.

3.2.2 Hierarchical Correction

We observe that the detector can perform at neighbor frames of trained frames with the same efficiency due to their similarity. Hence, to utilize the strength of the detector at neighbor frames, we propose a novel Hierarchical Correction module. Instead of correcting video frame sequentially, we propose correcting fixed key-frames, in which the frame-distance binarizedly decreases at each time step. The frame-distance D_k at iteration k is defined as $D_k = \left\lfloor \frac{D_{k-1}}{2} \right\rfloor$

where D_0 is the initial frame-distance. Human annotator easily interferes to correct mistakes at frame i, where $(i \bmod D_k) = 0$. The correction is then propagated to the next frame $i + D_k$, using SiamDW tracker [44].

After that, the correction of key-frames is fed back to train the detector for the next iteration to effectively guide the detector back to the right track. It guarantees for obtaining high-quality annotations while minimizing annotation time. Our Hierarchical Correction is more efficient than the standard frame-by-frame annotation, resulting in the faster interactive annotation.

4. Experimental Results

4.1. Implementation

All experiments were conducted on a computer with a Core i7 3.6 GHz processor, 64 GB of RAM, and two GTX 1080ti GPUs. We implement the code with Python and PyTorch.

For our detector, Faster R-CNN[2] [26] was adopted with several modifications. We applied Group Normalization [41] architecture, which uses ResNet-50 [11] based Feature Pyramid Network [21] (FPN) backbone. We also replaced the original ROI Pooling [26] by Precise RoI Pooling [13] to extract stronger FPN features from detected regions of interest (ROIs). Focal Loss [22] was applied to train Region Proposal Network [26] (RPN).

To train detectors, we used the Stochastic Gradient Descent (SGD) optimizer [5] with a moment of 0.9 and a weight decay of 0.0001. We trained our models with a batch size of 8. To obtain the pre-trained model, we trained our customized Faster R-CNN on MS-COCO [23] BDD [42] datasets with two schedules and the base learning rate of 0.02. During annotation processes (both ARA and IRA), detectors were fine-tuned on 0.5 schedules with the base learning rate of 0.002. At each training schedule, the model was trained on 25 epochs, and the base learning rate was divided by 10 at 8^{th}, 11^{th}, 16^{th} epochs.

For the ARA process, we repeated the automatic process in 3 iterations and practically set thresholds of Labeling Assistant as follow: $\tau_s = 0.85$, $\tau_t = 5$, and $\tau_{iou} = 0.3$. For the IRA process, the initial frame-distance D_0 of Hierarchical Correction is adaptively set based on the length of the video (T frames) to balance processing time and performance: for short videos, $D_0 = \frac{T}{12}$ where $T \leq 100$; for medium videos, $D_0 = \frac{T}{24}$ where $100 < T < 1000$; for long videos, $D_0 = \frac{T}{48}$ where $T \geq 1000$.

In this paper, we focus on annotation for challenging tasks of autonomous driving and intelligent transportation system, thus we evaluated the proposed framework on large-scale road object datasets (e.g. CityScapes [7] and DAD [6] datasets). We used seven popular object categories of moving objects on road, including *pedestrian*, *rider*, *car*, *truck*, *bus*, *motorbike*, and *bicycle*.

[2] https://github.com/facebookresearch/maskrcnn-benchmark

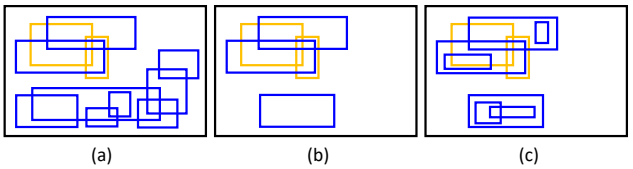

Figure 6. Our Similarity In Union (SIU) shows advantages against Average Precision (AP) and Intersection Over Union (IOU) in the task of annotation. Yellow bounding boxes are ground-truth and blue bounding boxes are detection results. From (a) and (b), AP cannot evaluate noise detection while SIU can. From (b) and (c), IOU cannot evaluate confident scores and overlapping regions while SIU can.

4.2. Evaluation Metric

Detection related metrics (i.e., Average Precision (AP) [9] and Average Best Overlap [36]) only focus on objects of interest without concerning the background. They ignore all noise detection, where detected bounding boxes do not touch ground-truth ones. Hence, these metrics are not appropriate to evaluate the task of annotation, which considers the global context. On the other hand, we observe that segmentation tasks (i.e., semantic segmentation [30], salient object segmentation [19, 17], camouflaged object segmentation [18]) always consider the whole scene, including both foreground (e.g., objects of interest) and background (e.g., noise detection). Hence, we leverage properties of segmentation to propose a new metric based on Intersection Over Union (IOU) to evaluate the task of object annotation.

We first convert all detection of an image to a heatmap of each category by accumulating the prediction score of each object region (i.e., bounding box or mask). Let $P^{(c)}$ and $Q^{(c)}$ denote heatmaps of prediction and ground-truth, respectively, of category $c \in C$, where C is all categories in the dataset. We introduce a new metric, namely Similarity In Union (SIU), to evaluate the similarity of two normalized heatmaps:

$$SIU^{(c)} = 1 - \frac{\sum_{i \in P^{(c)} \cup Q^{(c)}} |P_i^{(c)} - Q_i^{(c)}|}{|P^{(c)} \cup Q^{(c)}|} \quad (2)$$

Differently from Mean Absolute Error [1], which measures the difference of all pixels regardless of whether they do not belong to both prediction and ground-truth regions, our SIU skips these regions and measures only regions inside the union of prediction and ground-truth. We also note that IOU is a special case of our SIU, in which heatmaps are binary masks with only two values 0 and 1. Figure 6 illustrates the advantage of our SIU against AP and IOU.

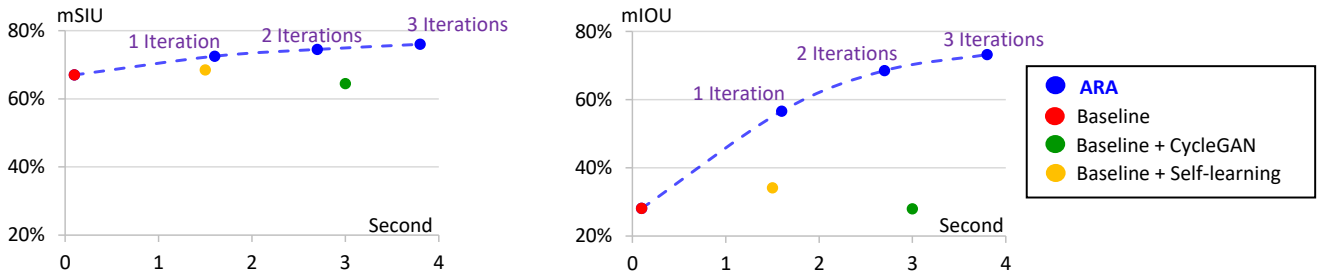

Figure 7. Experimental results on the Cityscapes [7] dataset using different metrics (mSIU and mIOU in the order from left to right). The processing time includes training adaptation methods and excludes training baseline. Our method, shown in blue, is converged after three iterations in both mSIU and mIOU.

Table 1. Detection results of Automatic Recurrent Annotation (ARA) on the Cityscapes [7] dataset.

Method	**Mean**	**Pedestrian**	**Rider**	**Car**	**Truck**	**Bus**	**Motorbike**	**Bicycle**
Similarity In Union	***mSIU***				***SIU***			
Baseline (Pre-trained model)	67.0	74.2	63.9	83.9	66.1	55.7	69.4	55.8
ARA (1 Iteration)	72.5	65.1	67.6	83.0	74.0	73.6	85.4	58.4
ARA (2 Iterations)	74.5	61.5	67.6	79.1	84.9	85.7	85.9	56.8
ARA (3 Iterations)	74.6	59.8	67.4	75.4	85.0	87.7	85.9	60.9
Intersection Over Union	***mIOU***				***IOU***			
Baseline (Pre-trained model)	28.1	15.4	55.8	20.1	1.6	4.3	66.4	33.4
ARA (1 Iteration)	56.6	36.2	65.5	37.2	71.5	68.0	84.7	32.8
ARA (2 Iterations)	68.5	47.1	66.8	66.4	84.6	84.8	85.2	44.6
ARA (3 Iterations)	73.2	55.0	67.3	76.8	84.8	87.3	85.4	55.6

Table 2. Ablation study on Cityscapes [7] dataset. Our method is shown in blue.

Method	**mSIU**	**mIOU**
Without Labeling Assistant	57.3	34.1
Only thresholding	73.6	69.5
With Labeling Assistant	74.6	73.2

Table 3. Interaction results on the DAD [6] dataset in terms of annotation time for a video frame and the ratio of correction by human annotator. Our method is shown in blue.

Method	**Annotation Time (Second)**	**Ratio of Corrected Frames**	**Ratio of Corrected Objects**
Manual	174	100%	100%
Pre-train+	129	100%	66.6%
ARA+	74	100%	45.7%
IRA	47	83.3%	61.3%
ISA	19	66.7%	35.9%

For multiple categories evaluation, SIU is averaged across categories, yielding the mean Similarity In Union (mSIU): $mSIU = \sum_{c \in C} SIU^{(c)}$.

4.3. Automatic Annotation Evaluation

We evaluated our automatic process (ARA) on the CityScapes [7] dataset. We used video sequences of the validation set, which consists of 500 sequences with a total of 15,000 video frames (each video sequence has 30 frames) from three cities in Europe (e.g., Frankfurt, Lindau, and Munster). The dataset has ground-truth at 20^{th} frame of each sequence. In addition to our introduced SIU, we also evaluate methods using IOU when binarizing heatmaps by assigning 1 for pixels whose values are larger than 0. Similarly to mSIU, mean Intersection Over Union (mIOU) is also averaged across all categories.

Iterations in The Loop. We study the ability of self-supervised learning of our ARA process in the loop by increasing the number of iterations. Figure 7 shows that detection results are converged from the third iteration, in which both mSIU and mIOU are around 74%. Hence, we stop the loop at the third iteration to balance processing time and performance. After the convergence, our ARA process achieves mSIU and mIOU in 74.6% and 73.2%, respectively. We also show the improvement of each category over the iterations in Table 1. Figure 8 visualizes the results of the ARA process on the CityScapes dataset.

Cross-Domain Evaluation. To measure the ability of our method in annotation, we performed the cross-domain evaluation for different methods. We consider the pre-trained model on the BDD dataset [42] as the baseline (denoted by *Baseline*). From this baseline model, we apply different methods to adapt domain from the BDD dataset to the CityScapes dataset, such as: Training CycleGAN [45] to transfer BDD-style images to Cityscapes-style images (denoted by *Baseline + CycleGAN*); Generating pseudo data by combining tracking and detection on the CityScapes

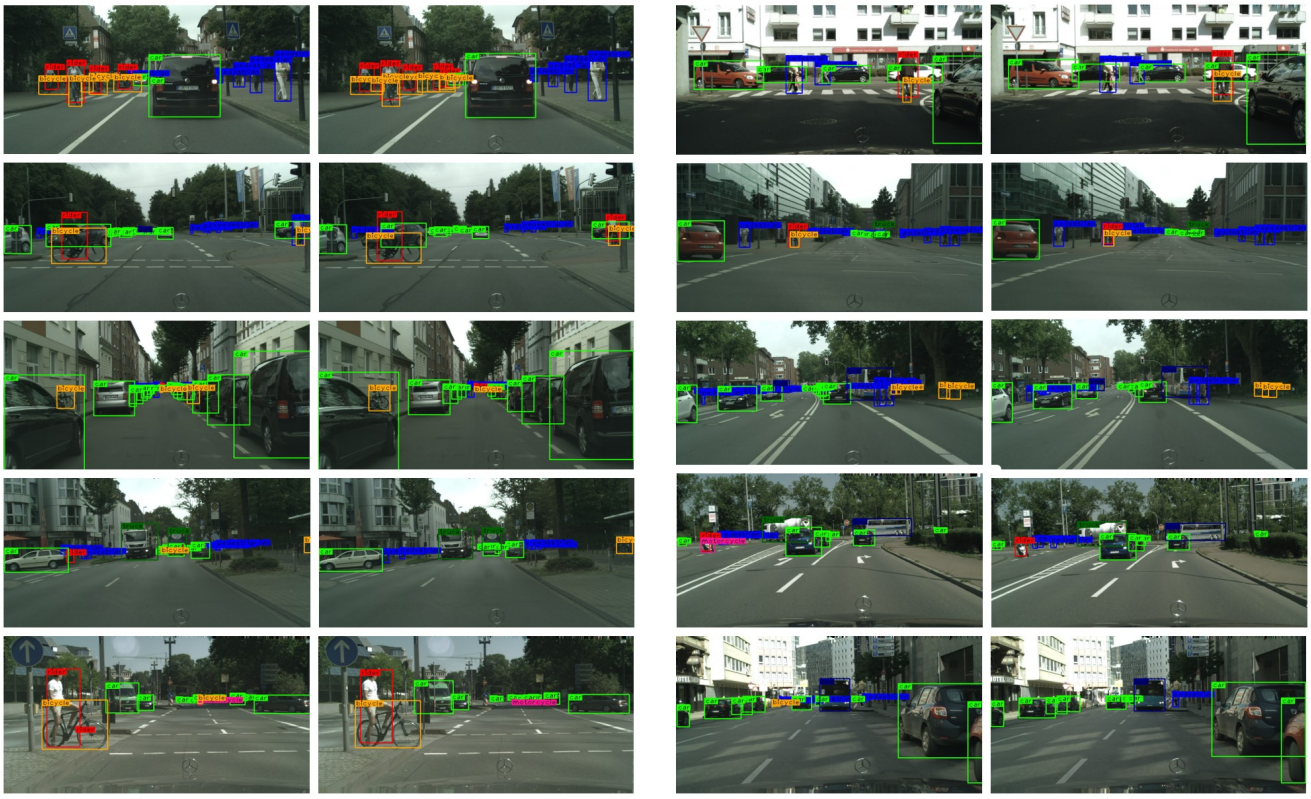

Figure 8. Visualization of some results by our ARA process on the CityScapes dataset. From left to right, original video frames with ground-truth are followed by results of our ARA process, respectively.

dataset for detector self-learning [27] (denoted by *Baseline + Self-learning*). For CycleGan [45], we used the published code provided by authors[3]. For self-learning [27], we re-implemented the method due to the unavailable published source code.

Figure 7 shows that our method outperforms other methods on all metrics. CycleGAN works only on small or medium resolution images while CityScapes videos have large resolution, leading to the reduction in the performance. Although our Labeling Assistant and label refinement of *Baseline + Self-learning*[27] share the same idea, implemented processes are different. Particularly, in our Labeling Assistant, after linking confident objects using [40], we remove short tubelets first, then assign a unique label for each tubelet, and finally extend tubelets using [44]. Meanwhile, in label refinement [27], after linking confident objects using [14], they extend tubelets using [24], and then remove short tubelets without considering label of objects (see Section 3.1 of [27]). The process of [27] causes many noise detection existing in training detector and duplicated objects having different labels. Hence, our ARA even using only one iteration outperforms *Baseline + Self-learning*[27] (72.5% and 68.5% of mSIU; 56.6% and 34.1% of mIOU, respectively).

Effect of Labeling Assistant. We investigate the effectiveness of the Labeling Assistant module by comparing our method against *without using Labeling Assistant* and applying *only threshold* to remove low confident objects. Table 2 shows the out-performance of our Labeling Assistant against other methods on all metrics. Some visualization results of Labeling Assistant are shown in Fig. 4.

4.4. Interactive Annotation Evaluation

We evaluated our proposed framework on the Dashcam Accident [6] (DAD) dataset. We used raw video sequences of the accident set, which consists of 620 sequences with a total of 62,000 video frames (each video sequence has 100 frames) from cities in Taiwan. We re-annotated all objects on all video frames, which have not been done in the published ground-truth. The link to our new ground-truth will be available along with the publication of this paper.

We evaluated our proposed framework *ISA* against other annotation methods such as: *Manual* annotation of all objects frame-by-frame; Using the pre-trained model to generate bounding boxes and then manually correct frames (denoted by *Pre-train+*); Correcting mistakes sequentially from results generated by ARA (denoted by *ARA+*); and

[3] https://github.com/junyanz/pytorch-CycleGAN-and-pix2pix

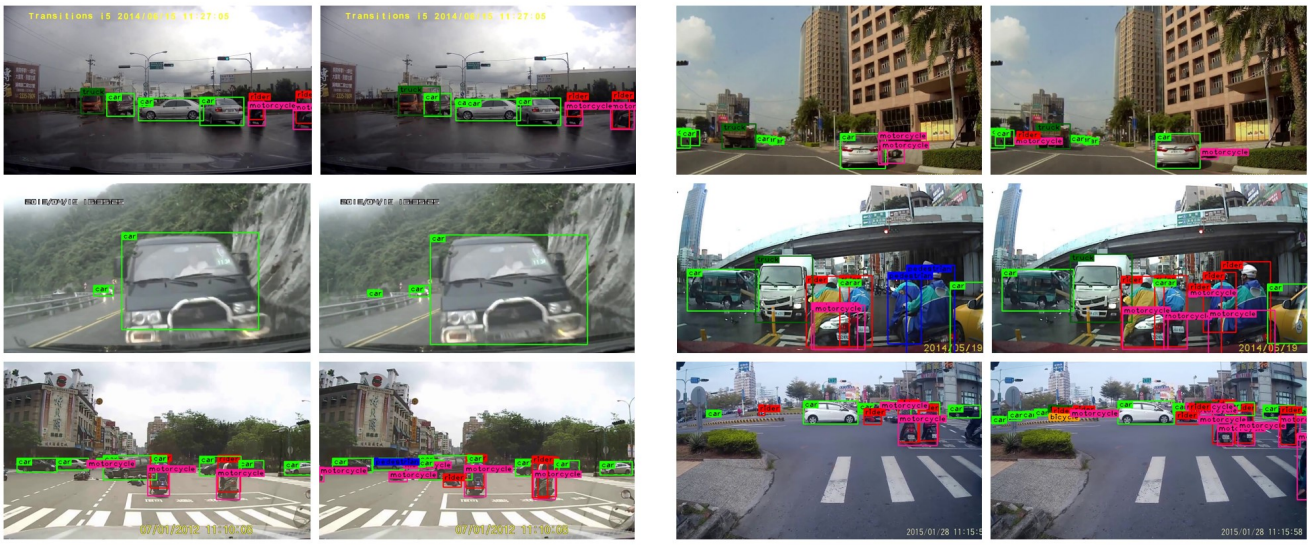

Figure 9. Visualization of some results by our ARA process and ISA framework on the DAD dataset in the order from left to right.

IRA directly using the pre-trained model as the initial detector. Table 3 illustrates experimental results.

Overall Evaluation. Experimental results in Table 3 shows that our proposed ISA framework is $9\times$ faster than manual annotation, that are 19 and 174 seconds to annotate a video frame, respectively. Particularly, the supported process ARA takes 4 seconds, in which training networks takes 3 seconds; and the main process IRA takes 15 seconds. In addition, our ISA is $6.8\times$ faster than annotating directly from the pre-trained model (19 seconds of ISA compared with 129 seconds of Pre-train+). Our ISA also cuts down the most human labor cost in terms of both the number of corrected frames (66.7%) and the number of corrected objects (35.9%). Our method minimizes both annotation time and human labor costs rather than the standard annotation method. Figure 9 illustrates the results of our ISA framework.

Effect of ARA Process. Our ISA, using the supported detector from ARA, is $2.5\times$ faster than IRA alone, which directly uses the pre-trained model as the initial detector (19 seconds and 47 seconds respectively). Integrating the ARA process also help ISA reduce the ratio of objects needing to be corrected by human annotator to 25.4% (from 61.3% to 35.9%). Furthermore, even not using our ISA framework but manually correcting results from the initial detector directly, ARA also speeds up the process 1.7 times and saves 20.9% number of corrected objects (comparing ARA+ and Pre-train+). This shows that the importance of our ARA process in training a robust model for video object annotation.

Effect of IRA Process. Our ISA framework is $3.9\times$ faster than ARA+, which does not integrate the IRA process. ISA takes only 19 seconds to annotate a video frame; meanwhile, ARA+ takes up to 74 seconds. By applying Hierarchical Correction, the IRA process also reduces the number of corrected frames to 66.6% by utilizing the strength of CNN and feedback from human annotator, comparing with correcting all video frames of ARA+. This highlights the impact of our IRA process in the interactive annotation.

5. Conclusion

In this paper, we proposed an Interactive Self-Annotation framework to minimize time and human labor costs to annotate bounding box of video objects. Our framework consists of two annotation processes: Automatic Recurrent Annotation, in which we let a detector watch unlabeled videos repeatedly to reinforce itself automatically, and Interactive Recurrent Annotation, which smoothly incorporates interactive correction from a human annotator in the loop to gradually improve the detector. We believe that our annotation framework will promote generating high-quality annotations while cutting down annotation time. We aim to consider annotating both bounding boxes and masks of video objects in the near future.

Acknowledgements

This research was, in part, supported by Committee on Advanced Road Technology (CART) 2018 of Ministry of Land, Infrastructure, Transport and Tourism (MLIT) of Japan and JSPS/KAKENHI 18H04119, 18K19824 of Ministry of Education, Culture, Sports, Science and Technology (MEXT) of Japan.

References

[1] R. Achanta, S. Hemami, F. Estrada, and S. Susstrunk. Frequency-tuned salient region detection. In *Conference on Computer Vision and Pattern Recognition*, pages 1597–1604, 2009.

[2] M. Bellver, A. Salvador, J. Torres, and X. G. i Nieto. Budget-aware semi-supervised semantic and instance segmentation. In *Conference on Computer Vision and Pattern Recognition*, June 2019.

[3] S. Bianco, G. Ciocca, P. Napoletano, and R. Schettini. An interactive tool for manual, semi-automatic and automatic video annotation. *Computer Vision and Image Understanding*, 131:88–99, 2015.

[4] T. A. Biresaw, T. Nawaz, J. Ferryman, and A. I. Dell. Vitbat: Video tracking and behavior annotation tool. In *International Conference on Advanced Video and Signal Based Surveillance*, pages 295–301, Aug 2016.

[5] L. Bottou. Large-scale machine learning with stochastic gradient descent. In *International Symposium On Computational Statistics*, pages 177–186, 2010.

[6] F.-H. Chan, Y.-T. Chen, Y. Xiang, and M. Sun. Anticipating accidents in dashcam videos. In *Asian Conference on Computer Vision*, pages 136–153, 2016.

[7] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. The cityscapes dataset for semantic urban scene understanding. In *Conference on Computer Vision and Pattern Recognition*, pages 3213–3223, 2016.

[8] D. Doermann and D. Mihalcik. Tools and techniques for video performance evaluation. In *International Conference on Pattern Recognition*, volume 4, pages 167–170, 2000.

[9] M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes challenge: A retrospective. *International Journal of Computer Vision*, 111(1):98–136, 2015.

[10] D. Feng, L. Rosenbaum, and K. Dietmayer. Towards safe autonomous driving: Capture uncertainty in the deep neural network for lidar 3d vehicle detection. In *International Conference on Intelligent Transportation Systems*, pages 3266–3273, 2018.

[11] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Conference on Computer Vision and Pattern Recognition*, pages 770–778, June 2016.

[12] B. L. J. Luiten, P. Voigtlaender. Premvos: Proposal-generation, refinement and merging for the davis challenge on video object segmentation. *Conference on Computer Vision and Pattern Recognition Workshops*, 2018.

[13] B. Jiang, R. Luo, J. Mao, T. Xiao, and Y. Jiang. Acquisition of localization confidence for accurate object detection. In *European Conference on Computer Vision*, pages 784–799, 2018.

[14] S. Jin, H. Su, C. Stauffer, and E. Learned-Miller. End-to-end face detection and cast grouping in movies using erdos-renyi clustering. In *International Conference on Computer Vision*, pages 5276–5285, 2017.

[15] M. Kabra, A. A. Robie, M. Rivera-Alba, S. Branson, and K. Branson. Jaaba: interactive machine learning for automatic annotation of animal behavior. *Nature methods*, 10(1):64, 2013.

[16] A. Khoreva, R. Benenson, E. Ilg, T. Brox, and B. Schiele. Lucid data dreaming for object tracking. *Conference on Computer Vision and Pattern Recognition Workshops*, 2017.

[17] T. Le and A. Sugimoto. Semantic instance meets salient object: Study on video semantic salient instance segmentation. In *IEEE Winter Conference on Applications of Computer Vision*, pages 1779–1788, Jan 2019.

[18] T.-N. Le, T. V. Nguyen, Z. Nie, M.-T. Tran, and A. Sugimoto. Anabranch network for camouflaged object segmentation. *Journal of Computer Vision and Image Understanding*, 184:45–56, 2019.

[19] T.-N. Le and A. Sugimoto. Deeply supervised 3d recurrent fcn for salient object detection in videos. In *British Machine Vision Conference*, 2017.

[20] L.-J. Li, R. Socher, and L. Fei-Fei. Towards total scene understanding: Classification, annotation and segmentation in an automatic framework. In *Conference on Computer Vision and Pattern Recognition*, pages 2036–2043, 2009.

[21] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie. Feature pyramid networks for object detection. In *Conference on Computer Vision and Pattern Recognition*, 2017.

[22] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár. Focal loss for dense object detection. In *International Conference on Computer Vision*, pages 2980–2988, 2017.

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

[24] H. Nam and B. Han. Learning multi-domain convolutional neural networks for visual tracking. In *Conference on Computer Vision and Pattern Recognition*, pages 4293–4302, 2016.

[25] K.-T. Nguyen, T.-H. Hoang, M.-T. Tran, T.-N. Le, N.-M. Bui, T.-L. Do, V.-K. Vo-Ho, Q.-A. Luong, M.-K. Tran, T.-A. Nguyen, T.-D. Truong, V.-T. Nguyen, and M. N. Do. Vehicle re-identification with learned representation and spatial verification and abnormality detection with multi-adaptive vehicle detectors for traffic video analysis. In *Conference on Computer Vision and Pattern Recognition Workshops*, 2019.

[26] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems*, pages 91–99, 2015.

[27] A. RoyChowdhury, P. Chakrabarty, A. Singh, S. Jin, H. Jiang, L. Cao, and E. Learned-Miller. Automatic adaptation of object detectors to new domains using self-training. In *Conference on Computer Vision and Pattern Recognition*, pages 780–790, 2019.

[28] B. C. Russell, A. Torralba, K. P. Murphy, and W. T. Freeman. Labelme: a database and web-based tool for image annotation. *International journal of computer vision*, 77(1-3):157–173, 2008.

[29] B. Sekachev, N. M., and A. Z. Computer vision annotation tool: A universal approach to data annotation, 2019. https://software.intel.com/en-us/articles/computer-vision-annotation-tool-a-universal-approach-to-data-annotation.

[30] E. Shelhamer, K. Rakelly, J. Hoffman, and T. Darrell. Clockwork convnets for video semantic segmentation. In *European Conference on Computer Vision*, pages 852–868, 2016.

[31] K. K. Singh and Y. J. Lee. You reap what you sow: Using videos to generate high precision object proposals for weakly-supervised object detection. In *Conference on Computer Vision and Pattern Recognition*, June 2019.

[32] O. Styles, A. Ross, and V. Sanchez. Forecasting pedestrian trajectory with machine-annotated training data. In *Intelligent Vehicles Symposium*, June 2019.

[33] H. Su, J. Deng, and L. Fei-Fei. Crowdsourcing annotations for visual object detection. In *AAAI Conference on Artificial Intelligence Workshops*, 2012.

[34] M.-T. Tran, T.-N. Le, T. V. Nguyen, V. Ton-That, T.-H. Hoang, N.-M. Bui, T.-L. Do, Q.-A. Luong, V.-T. Nguyen, D. A. Duong, and M. N. Do. Guided instance segmentation framework for semi-supervised video instance segmentation. *Conference on Computer Vision and Pattern Recognition Workshops*, 2019.

[35] M.-T. Tran, V. Ton-That, T.-N. Le, K.-T. Nguyen, T. V. Ninh, T.-K. Le, V.-T. Nguyen, T. V. Nguyen, and M. N. Do. Context-based instance segmentation in video sequences. *Conference on Computer Vision and Pattern Recognition Workshops*, 2018.

[36] J. R. Uijlings, K. E. Van De Sande, T. Gevers, and A. W. Smeulders. Selective search for object recognition. *International Journal of Computer Vision*, 104(2):154–171, 2013.

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

[38] B. Wang, C. Zheng, N. Wang, S. Wang, X. Zhang, S. Liu, S. Gao, K. Lu, D. Zhang, L. Shen, Y. Wang, and Y. Xu. Object-based spatial similarity for semi-supervised video object segmentation. *Conference on Computer Vision and Pattern Recognition Workshops*, 2019.

[39] S. Wang, Y. Zhou, J. Yan, and Z. Deng. Fully motion-aware network for video object detection. In *European Conference on Computer Vision*, pages 542–557, 2018.

[40] N. Wojke, A. Bewley, and D. Paulus. Simple online and realtime tracking with a deep association metric. In *International Conference on Image Processing*, pages 3645–3649, 2017.

[41] Y. Wu and K. He. Group normalization. In *European Conference on Computer Vision*, pages 3–19, 2018.

[42] F. Yu, W. Xian, Y. Chen, F. Liu, M. Liao, V. Madhavan, and T. Darrell. Bdd100k: A diverse driving video database with scalable annotation tooling. *arXiv:1805.04687*, 2018.

[43] J. Yuen, B. Russell, C. Liu, and A. Torralba. Labelme video: Building a video database with human annotations. In *International Conference on Computer Vision*, pages 1451–1458, 2009.

[44] Z. Zhang and H. Peng. Deeper and wider siamese networks for real-time visual tracking. In *Conference on Computer Vision and Pattern Recognition*, June 2019.

[45] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *International Conference on Computer Vision*, 2017.