A Delay Metric for Video Object Detection: What Average Precision Fails to Tell

Huizi Mao
Stanford University
huizimao@stanford.edu

Xiaodong Yang
NVIDIA
xiaodongy@nvidia.com

William J. Dally
Stanford University & NVIDIA
dally@stanford.edu

Abstract

Average precision (AP) is a widely used metric to evaluate detection accuracy of image and video object detectors. In this paper, we analyze object detection from videos and point out that AP alone is not sufficient to capture the temporal nature of video object detection. To tackle this problem, we propose a comprehensive metric, average delay (AD), to measure and compare detection delay. To facilitate delay evaluation, we carefully select a subset of ImageNet VID, which we name as ImageNet VIDT with an emphasis on complex trajectories. By extensively evaluating a wide range of detectors on VIDT, we show that most methods drastically increase the detection delay but still preserve AP well. In other words, AP is not sensitive enough to reflect the temporal characteristics of a video object detector. Our results suggest that video object detection methods should be additionally evaluated with a delay metric, particularly for latency-critical applications such as autonomous vehicle perception.

1. Introduction

There is a growing interest in video object detection. Many real-world applications, such as surveillance analysis and autonomous driving, deal with video streams. Several single-image object detection algorithms have been proposed in the past few years [5, 20, 30], but they are compute-intensive to run on a full-resolution video stream. Exploiting temporal information is therefore an important direction to improve the accuracy-cost trade-off [12, 19, 33].

Prior research suffers the lack of densely annotated video datasets. KITTI [9] is a dataset targeting at autonomous driving that provides frame-level bounding box annotations. However, it is relatively small compared with other large-scale datasets for training deep neural networks. Since the introduction of object detection from video challenge (VID) [6], more research focus has been drawn into the study of video object detection algorithms.

There are two general goals of video object detection: improving detection accuracy [2, 8, 12, 37] and reducing computational cost [4, 24, 38]. Currently, the accuracy of proposed detection algorithms are mostly evaluated with average precision (AP) or mean average precision (mAP) that is the average of APs over all classes [6, 9, 18]. Video object detection benchmarks like VID also adopt the mAP, where every frame is treated as an individual image for evaluation. However, such an evaluation metric ignores the temporal nature of videos and fails to capture the dynamics of detection results, e.g., a detector that detects the later half occurrences of an instance holds the same mAP as a detector that detects every other frame. As indicated in later experiments, video detectors tend to demonstrate different temporal behaviors compared to their single-image counterparts.

We introduce average delay (AD), a new detection delay metric. Measuring video object detection delay seems trivial, as the delay can be simply defined as the number of frames from when an object appears to when it is detected. However, to avoid the case where an algorithm trivially detects every bounding box in an image, a false alarm rate constraint is necessary. AD also needs to be designed to be comprehensive like AP, so that the delays at different false alarm rates can be combined. We discuss our design rationale in Section 3.

Most video snippets in VID contain fixed numbers of instances (typically only one), which is not suitable for the delay evaluation. We therefore select a portion of the validation set in VID and name it as VID with multiple tracklets (VIDT). Details of the new VIDT dataset are described in Section 4. With VIDT we then evaluate the AD of a wide range of the recent proposed video detection algorithms in Section 5. A general trend is shown in Figure 1, which indicates that some computation-reducing methods [24, 38] preserve the mAP well but increase the AD. Alternative methods leverage the temporal information to improve detection accuracy but worsen the detection delay [37]. Our results suggest that video object detection methods should be evaluated with a delay metric, particularly for latency-critical applications such as autonomous vehicle perception.

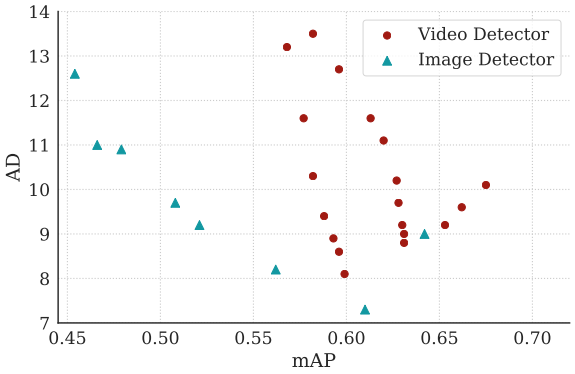

Figure 1. AD does not strongly correlate with mAP. Many algorithms that are specifically designed for video object detection fail to achieve similar AD as the frame-by-frame image detectors, although they may have higher mAP. Image object detectors include R-FCN, Faster R-CNN and RetinaNet. Video object detectors include DFF, FGFA and CaTDet.

To our knowledge, this is the first work that brings up and compares detection delay, a highly critical but usually ignored issue, for video object detection. We propose a comprehensive evaluation metric AD to measure and compare video object detection delay[1]. By evaluating a variety of video object detection algorithms, we analyze the key factors for detection delay and provide the guidance for future algorithm design.

2. Background

2.1. Overview of Video Object Detection

Video object detection performs a similar task as image object detection, except that the former is carried on a video stream. Densely annotated videos, which are costly to obtain, are typically required to train a video object detector. The ImageNet VID challenge greatly advances the research progress in the field of video object detection, and provides a large frame-by-frame annotated dataset that covers a wide range of scenarios.

Various methods have since been proposed and evaluated on the VID dataset. The goal of video object detection is to reduce the computational cost or refine the detection results by exploiting the temporal dimension of videos. For instance, deep feature flow (DFF) [38], detect or track (DorT) [21], CaTDET [24] and saptiotemporal sampling networks [2] fall into the first category, while T-CNN [12], detect to track (DtoT) [8] and LSTM-aided SSD [19] belong to the second category. These methods are typically variants of the well studied image object detection algorithms such as R-FCN [5], Faster R-CNN [30], SSD Multibox [20] and RetinaNet [17].

[1]Code available at https://github.com/RalphMao/VMetrics.

As required in the VID challenge, the performance of a video object detector is solely evaluated by mAP, the metric for still image object detection [6, 7, 18]. When evaluating mAP, every single frame of a video is treated as an individual image. In such a way, the quality of a detector over the whole video sequence is measured and compared.

2.2. Low Latency as a Practical Requirement

Low latency is a common requirement for many video-related applications. For example, autonomous driving typically requires less than 100ms latency [16]. Detecting an object with minimum delay is desired, and detection after certain time is no longer important.

In previous research, the term latency mostly refers to computational latency only [1, 26]. However, we argue that the overall latency equals to **computational latency** plus **algorithmic delay**, and the latter is the time taken in a video stream for an algorithm to finally determine the existence of an object. Computational latency has been extensively studied in the recent works [11, 25, 36], while algorithmic delay remains less explored in the object detection field. In other fields like activity detection, there have been efforts to study early detection [22].

2.3. Relevant Studies on the Delay Issue

Quickest change detection (QCD) is a well studied problem in statistical processing. It refers to real-time detection of abrupt changes in the behavior of an observed signal or time series as quickly as possible [28]. Generally, the delay is measured at a certain constraint of false alarms. Lao et al. [14] targeted the problem of moving object detection at a minimum delay. They formulated the task under the framework of QCD and gave an optimal solution for the single object case.

NAB [15] is a benchmark for real-time anomaly detection in time-series data. The authors pointed out that traditional scoring methods such as precision and recall do not suffice, as they cannot effectively test anomaly detection algorithms for real-time use. To reward early detection, they define anomaly windows. Inside the window, true positive detections are scored by a sigmoid function and out of the window all detections are ignored.

In the field of video action recognition [23, 32, 34], early detection has also gained attention [13, 31]. This task typically requires accumulating enough frames to make a decision. To alleviate this issue, a special loss function was proposed to encourage early detection of an activity [22].

All of the works above are essentially dealing with the single object or single signal case. CATDet [24] introduced a delay metric to measure the detection delay for multiple objects. However, the delay is evaluated at a specific precision only to counter false alarms.

3. The Average Delay Metric

In this section, we present our definition of average delay (AD), the evaluation metric for video object detection delay. Our metric is designed to incorporate fairness and comprehensiveness. **Fairness**: AD considers the trade-off between false positives and false negatives to avoid the case of reducing delay by detecting many false positives. **Comprehensiveness**: AD covers a wide range of operating conditions, analogous to AP.

We first explain the terminology used throughout this paper before delving into the detailed derivation. An *instance* is a physical object that appears in consecutive frames as a trajectory (or a tracklet). An *object* refers to a single occurrence of an instance in a frame. The ground truth of an object includes its bounding box coordinates, class label, and track identity. A *detection* is the recognition of an object in one frame with bounding box coordinates, class label, and confidence.

3.1. Delay and Statistical Process of Detection

The most intuitive definition of delay is the number of frames taken to detect an instance from the frame it appears. Before reasoning on a comprehensive delay metric, we make this simple assumption: a detector detects every object at every frame with the same probability p.

Under this assumption, the delay D follows the discrete exponential distribution: $D \sim \exp(p)$. Figure 2 exemplifies a histogram of the detection delays of R-FCN on VIDT. The actual distribution generally resembles the exponential distribution, apart from an anomalous region in the tail. There are substantially more instances than expected with extremely large delays due to existence of "hard instances". A detailed discussion about the delay statistics is described in Section 6 and hard examples are given in Figure 10.

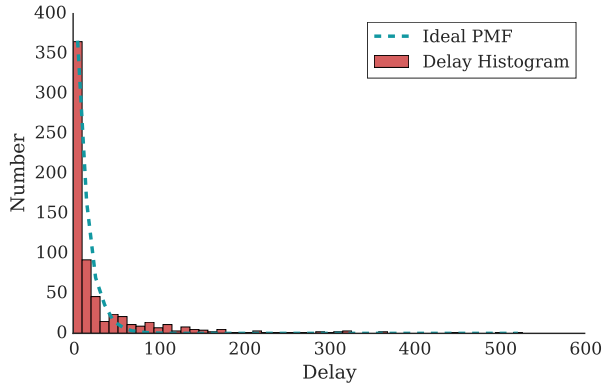

Figure 2. A delay histogram of R-FCN (ResNet-101) on VIDT at a confidence threshold of 0.5. We also show a plot of probability mass function (PMF) under an ideal discrete exponential distribution as the reference to the actual delay dstribution.

For discrete exponential distribution, the expected value follows $E(D) = 1/p - 1$. Thus, we can measure the quality of a detector through inferring the latent parameter p, given multiple observed datapoints D_i, where $i = 1, ..., N$. With maximum likelihood estimation, we find that the maximum likelihood is achieved when the expected value matches the mean of the samples: $E(D) = \frac{1}{N}\sum_{i=1}^{N} D_i = \bar{D}$. So the detection probability p on each frame can be obtained by:

$$p = \frac{1}{\bar{D} + 1} \tag{1}$$

As aforementioned, the existence of "heavy tail" results in a potential problem when we try to estimate p. Different detectors may not be effectively differentiated if the heavy tail dominates the mean value. We thus adopt a simple strategy to clip the delay samples with a constant value W, which we name as a *detection window*. This is also a practical consideration, as for most latency-critical tasks a detection no longer matters once it falls out of a time window.

$$\begin{aligned} p &= \frac{1}{\bar{D}^* + 1}, \\ \bar{D}^* &= \frac{1}{N}\sum_{i=1}^{N} \min(D_i, W). \end{aligned} \tag{2}$$

3.2. Choice of False Positive Ratio

It is important to set a threshold for false alarms to ensure fair comparisons. In the previous work [24], precision that is defined as number of true positives divided by total detections is selected as a threshold to counter false alarms, as the increased number of false alarms will reduce precision. However, there are undesired outcomes if we set the same precision to compare different detectors.

We demonstrate with a toy example in Figure 3 to illustrate that setting a precision threshold may cause the

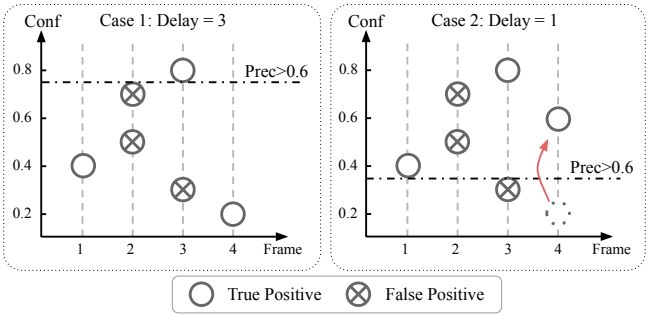

Figure 3. A toy example to illustrate that using precision as the control may lead to undesired behaviors. There is one ground truth instance in frames 1-4. We set the control as Prec > 0.6. Due to a more confident true positive at frame 4, case 2 has an unreasonable lower delay than case 1. Setting false positive ratio as the control would avoid this problem.

Figure 4. Snippets in the validation set of VID. Top: an ideal video snippet for delay evaluation with multiple instances emerging randomly over space and time. Bottom: an undesired video snippet, in which there is the same instance throughout the time.

measured delay to behave differently from our expectation. Suppose the precision threshold is set to 0.6. In case 1, we should set a confidence threshold of 0.75 to meet the precision requirement. In case 2, due to the increased confidence score of the last detection, a confidence threshold of 0.35 is adequate. The resulted detection delay in these two cases are 2 and 0, respectively. By refining the later detections, the detection delay can be magically improved. Such a behavior counters our intuition that delay should be a matter of the early detections.

As a result, we argue that precision may not be the ideal threshold to counter false alarms. Instead we propose to use false positive (FP) ratio, which is the ratio between false positives and ground truth objects. FP ratio as a threshold is determined only by false positive detections, therefore will not be impacted with more true positives.

3.3. A Comprehensive Metric

The last question comes that how we should comprehensively measure the detection delay of a detector under different false alarm constraints, similar to what AP does. AP is the integral or the arithmetic mean of precisions over different recalls. Analogously, is it a good practice to average detection delays over different false positive ratios?

Consider the real-world scenarios, a detector with zero delay is substantially better than one with 1-frame delay, while a detector with 14-frame delay does not make a significant difference from one with 15-frame delay. However, the arithmetic mean cannot distinguish the two cases.

We argue that averaging the latent parameter p, which represents the probability of detecting an object, would be a better choice. Since p is the reciprocal of $\bar{D}+1$, it weighs more for a smaller delay. In addition, it is a bounded value between 0 and 1. As a result, we average the inferred p values of a detector under different false positive ratios, and derive the corresponding AD fom the averaged $\bar{p}$.

We show our definition of the proposed AD in Equation 3. Here R stands for the total number of FP ratios and $\bar{D}_r^*$ is the delay measured by Equation 2 at a specific FP ratio r. Notice that this definition has a very similar form to the harmonic mean.

$$AD = \frac{1}{\bar{p}} - 1 = \frac{1}{\frac{1}{R}\sum_r \frac{1}{\bar{D}_r^*+1}} - 1 \tag{3}$$

In our following experiments, we set the detection window W to 30 frames and select 6 FP ratios including 0.1, 0.2, 0.4, 0.8, 1.6 and 3.2.

4. Dataset for Delay Evaluation

4.1. Overview

There are multiple public datasets for object detection, such as KITTI [9], ImageNet-VID [6], YouTube-BB [29], BDD100K [35], VIRAT [27], etc. However, they suffer from various drawbacks for delay evaluation. KITTI is a relatively small dataset, making it hard to train deep neural networks. Most video snippets in ImageNet VID contain fixed numbers of objects from beginning to end, which leaks strong prior, thus making it unsuitable for delay evaluation. Youtube-BB and BDD100K are both large-scale datasets with rich objects and scenarios, but they are sparsely annotated. VIRAT is a surveillance analysis dataset and has a fixed background.

An ideal dataset for delay evaluation should (**i**) be densely annotated (frame by frame); (**ii**) have random entry time for each instance (exclude videos with the same objects throughout the time); (**iii**) have random entry location for each instance (exclude videos with a fixed background and limited entry locations for new objects). In Figure 4, we show examples of ideal and non-ideal snippets in the validation set of ImageNet VID. The ideal snippet has multiple different instances entering the frames randomly over space and time, while in the non-ideal case, the same instance (which is a boat in the example) exists from the very first frame to the last.

4.2. Introducing VIDT

We introduce VIDT, a subset of the validation set of VID, to meet the requirements aforementioned. Video snip-

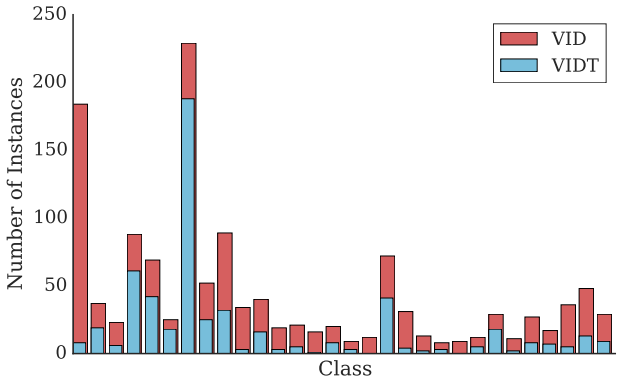

Figure 5. Number of instances per class is highly imbalanced in VID and VIDT. The class "Car" has most instances in both datasets. There is no instance of "Lizard" and "Sheep" in VIDT.

Dataset	Snippets	Frames	Instances	Objects
VIDT	120	53K	666	102K
VID-val	555	176K	1309	274K
VID-train	3862	1122K	7911	1732K
KITTI*	21	8K	783	41K

Table 1. Statistics of the candidate datasets for delay evaluation. Note that KITTI does not have an official split of train/val.

pets in VIDT have at least one instance entering at a non-first frame, which guarantees the randomness of entry time. VIDT largely relies on the annotated track identities in VID. A subtle difference is that in VIDT, once an instance disappears for more than 10 consecutive frames, it is marked as a new instance. One reason is that we do not care about the re-identification ability but only the capability to detect as early as possible. In this way, the number of instances is increased from 555 to 666.

We report the statistics of VIDT and compare with the original VID and KITTI in Table 1. The ample training data of VID makes it feasible to train deep neural networks. Even though VIDT is smaller than the original validation set of VID, it still has much more frames and objects than KITTI with training and validation sets combined. However, severe class imbalance problem exists in both VID and VIDT as shown in Figure 5. Therefore, AD is not measured on each class separately, but instead treats all instances in a class-agnostic way.

5. Experiments

In this section, we demonstrate a common but mostly ignored problem in recent research of video object detection. Many detectors suffer from worse detection delay, even though they are able to preserve or even improve mean Average Precision.

5.1. Toy Cases for Metric Comparison

We design several special cases to show the advantages of our proposed average delay metric against mAP [7], NAB Score [15] and CaTDet Delay [24]. The NAB metric, originally designed for anomaly detection, can be modified to fit in the object detection task. The modification is described in Appendix.

Our comparison of the different metrics is achieved by manipulating the detection output and quantifying the impact on each metric. Retardation measures the **sensitivity** by suppressing the first few detections of a tracklet. A desirable delay metric should be worsened after retardation. Tail boost measures the **fairness** by elevating the confidence scores of lately detected objects. A fair delay metric should not be affected by tail boost. Multiple observations can be drawn from Table 2.

- For mAP, retardation makes little impact while tail boost greatly improves the result, which is in accordance to the number of affected detections.
- Retardation worsens all three delay metrics. However, if suppressing the low-confident objects only, NAB and CaTDet do not reflect the change, as both of them operate at a single confidence threshold. In contrast, AD evaluates multiple thresholds, therefore is robust to reflect the effect of retardation.
- For tail boost, it improves both NAB and CaTDet, while only improves AD negligibly, indicating that AD is better than the other two metrics in term of fairness.

5.2. Key Frame based Methods

A range of recent works on video object detection employ the concept of the key frame [4, 10, 21, 38]. Key frames are sparsely distributed over the whole video sequence and typically require more computational resources

	Baseline	Retardation		Tail Boost
		Low-Conf	All	
# of affected detections	0	3076	3616	71781
mAP	0.64	0.63	0.63	0.70
NAB	0.29	0.29	0.17	0.31
CaTDet	13.6	13.6	15.1	12.5
AD (Ours)	9.0	11.5	13.8	8.9

Table 2. Comparison of the different metrics by data manipulation. Baseline is an R-FCN detector with ResNet-101. Retardation makes detection slower by suppressing the first 5 detections of a ground truth instance. In the case low-conf, we only suppress the detections with low confidence, while in the case all, all detections are suppressed regardless of their confidence scores. Tail boost improves the detections that are 20 frames later than the first occurrence of ground truth. Note that for CaTDet and AD, lower numbers indicate better results.

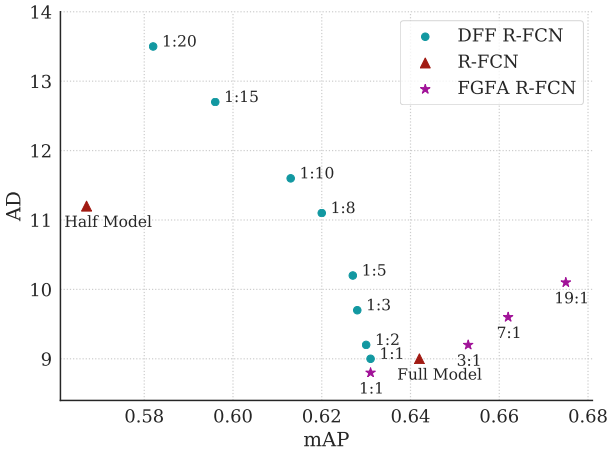

Figure 6. How DFF and FGFA affect mAP and AD. Here $1:N$ refers to 1 key frame in every N frames for DFF. $N:1$ refers to N frames aggregated for FGFA. The full and half models are both frame-by-frame R-FCN models, except that the half model is trained with half number of iterations. All models use ResNet-101 as the backbone.

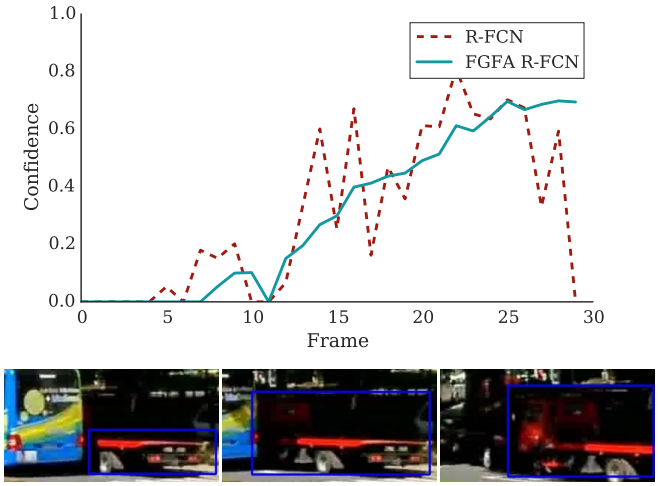

Figure 7. An example illustrates how FGFA causes higher detection delay. The frame-by-frame R-FCN model exhibits large fluctuation of confidence, while the FGFA model tends to slowly build up the confidence over time.

than non-key frames. Key frames can be used to improve the detection accuracy or reduce the cost of non-key frames, through exploiting the temporal locality in videos.

We choose deep feature flow (DFF) [38] as a representative key frame based algorithm. The basic idea is to compute features on key frames and propagate the features with optical flow on non-key frames. We vary the interval of key frames and show the impact on mAP and AD in Figure 6. Two R-FCN models are also reported for comparison. The full model is a standard R-FCN model with ResNet-101, and the half model is in the same architecture but trained with only half number of iterations.

Figure 6 shows that DFF tends to worsen AD. For example, the DFF model that adopts a key frame in every 10 frames achieves mAP of 0.613, much higher than the mAP 0.567 of the inferior R-FCN model. However, in term of AD, the DFF model is a bit worse (11.6 vs. 11.2). This indicates that setting sparse key frames leads to the delayed detection of new objects.

5.3. Feature Aggregation Methods

Combining features of multiple frames is an effective approach to improve detection accuracy. Recent works in the field include explicit feature aggregation by temporally adding up features [2, 37] and implicit feature aggregation via recurrent neural networks [19].

We select the flow-guided feature aggregation (FGFA) [37] as an example and demonstrate how it may affect detection delay while improving mAP. FGFA aggregates the features of previous frames and solves the spatial misalignment by propagating the features with optical flow. The open-sourced version of FGFA is based on R-FCN, therefore we also compare its mAP and AD in Figure 6. FGFA alone improves mAP from 0.642 to 0.675, meanwhile deteriorates AD from 9.0 to 10.2. We also observe a trend that the more frames aggregated, the better mAP can be obtained but the worse AD is.

FGFA substantially improves mAP compared with the original R-FCN, but worsens the detection delay. To explain this phenomenon, we select one instance with increased delay and plot the process of being detected in Figure 7, which shows the confidence score of the closest detection to the ground truth object. In the case where no detection has an IoU over 50%, the confidence score is 0. The steady and progressive increasing confidence of FGFA, as shown in the figure, incurs the extra delay to detection, suggesting that for latency-critical tasks it is probably not a good choice to slowly build up the confidence.

5.4. Cascaded Detectors

A cascaded detector consists of multiple components and tries to shift the workload from complex ones to simple ones, following specific heuristics. Bolukbasi et al. [3] proposed a selective execution model for object recognition problem, which is essentially a cascaded system. Further works explored the efficacy of cascaded systems in the video object detection task, including scale-time lattice [4] and CaTDet [24].

We adopt CaTDet [24] as an example. CaTDet adds a tracker in the cascaded model to enable temporal feedback, which helps save the workload and improve accu-

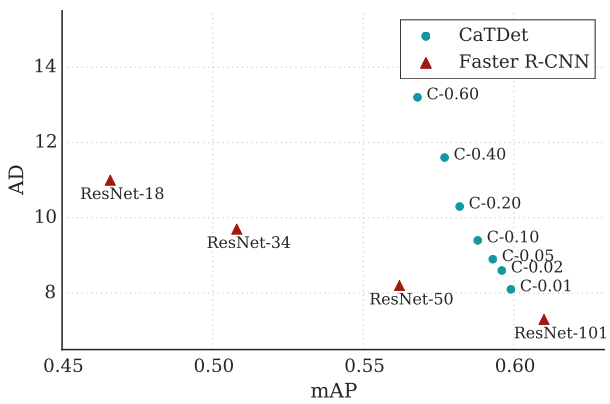

Figure 8. CaTDet preserves mAP well but incurs more AD, compared with Faster R-CNN with smaller models. C-α stands for CaTDet with an intermediate threshold of α. Larger α value saves more computation at the cost of more accuracy loss. All CaTDet models are based on Faster R-CNN with ResNet-101.

racy. As shown in Figure 8, CaTDet models preserve the mAP well but substantially increases detection delay compared with other Faster R-CNN models. The CaTDet model with an internal confidence threshold of 0.01 achieves mAP of 0.555, which is very close to that of the Faster R-CNN model (0.561), however, it increases AD from 8.2 to 9.2.

6. Analysis of Delay

In this section, we analyze the characteristics of video object detection delay on the VIDT dataset and aim to provide our insights into the AD metric.

6.1. Delay Distribution

In Section 3, we make an assumption that video object detection delay follows the discrete exponential distribution but with a heavy tail. Here we provide more examples and analysis to examine the actual distribution of delay.

We select the three object detection methods: R-FCN, Faster R-CNN and DFF R-FCN, and plot their delay distribution in Figure 9. All three distributions resemble the

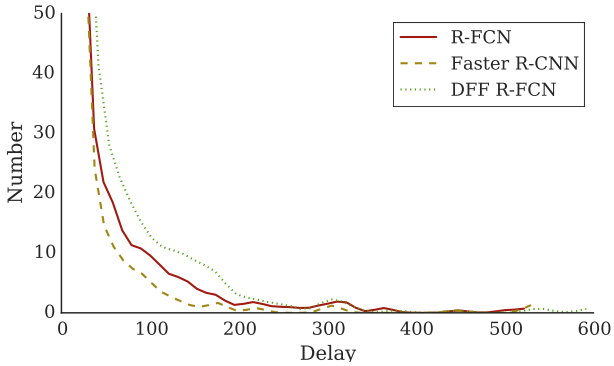

Figure 9. A zoomed-in plot of delay distribution of multiple detectors. All three models are based on ResNet-101 and have the same confidence threshold of 0.5. DFF runs with 1 key frame out of 10.

exponential distribution. Note that at the same confidence threshold Faster R-CNN has the smallest delay, therefore its delay distribution is more skewed to left compared with the other two approaches.

We also show the statistics to measure the "heavy-tail" effect in Table 3. The difference between mean and clipped mean denotes that the long tail has a large impact on the mean value. Here we define "expected off-window percentage" as the probability of the delay D falling out of a detection window, where D is assumed to follow the ideal discrete exponential distribution. The ideal distribution is estimated by the maximum likelihood estimation. Such a probability can be computed by $P = (1-p)^W$, where p is obtained as in Equation 1 and W is the window size. The higher percentages outside the window in all three detectors validate that the tails are indeed "heavier" than those in the ideal exponential distributions. We select 6 examples out of

	R-FCN	Faster R-CNN	DFF R-FCN
Mean	33.5	17.8	43.3
Clipped Mean	24.4	13.8	31.5
Off-Window Percentage	10.2%	3.6%	14.3%
Expected Off-Window Percentage	5.3%	0.4%	10.2%

Table 3. Statistics to show the heavy-tail effect of delay distribution: more than expected detections that exceed a 100-frame window. Clipped mean is the mean value computed with Equation 2.

Figure 10. Examples of hard instances that have larger than 100-frame delay for R-FCN with ResNet-101. All crops are warped into the same dimensions. They represent some typical cases that tend to result in large detection delay: low resolution (left), severely occluded (mid), blurry and occluded (right).

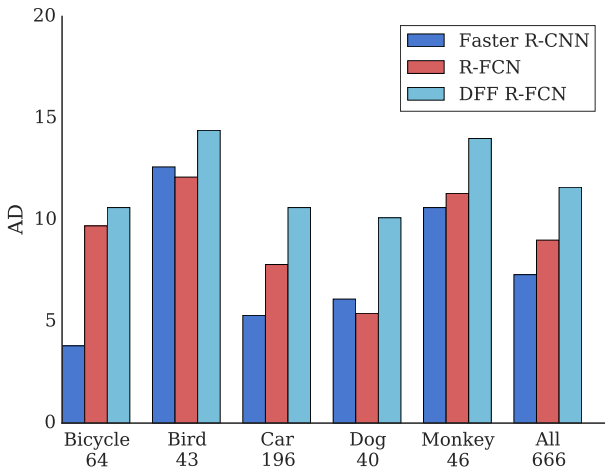

Figure 11. AD by class: we only demonstrate the six video object classes, each of which contains more than 40 instances. The number of instances is shown under each class name. All three detectors adopt ResNet-101 as the base model. DFF operates with 1 key frame out of 10.

10 with largest delay and illustrate them in Figure 10. These video objects are either with very low resolution, heavily truncated or largely occluded.

6.2. Average Delay of Different Classes

Due to the class imbalance in VIDT, AD is measured on all 666 instances instead of individual classes to avoid high variance. To demonstrate how the delay varies on different classes, we select 5 classes with over 40 instances and compare their AD results in Figure 11. All three models present large delays for class "Bird", which is typically small and quick moving. Classes "Car" and "Dog" have relatively smaller delays. For class "Bicycle", R-FCN and Faster R-CNN show distinct delays.

6.3. Average Delay of Different Scales

To study how the size of instances affects detection delay, we divide all 666 instances into 3 categories by the averaged shorter dimension D_s of their first 30 frames. *Small*, *Median* and *Large* instances are categorized according to $D_s < 40$, $40 \leq D_s < 100$ and $D_s \geq 100$. This criterion results in 129 small instances, 257 median instances and 280 large instances, respectively.

The anchor scale is the size of reference bounding box in all major object detection algorithms, where 3 and 4 scales are common choices for image object detection. As shown in Table 4, further increasing the number of anchor scales from 3 to 4 does not improve mAP. However, adding a small scale helps with AD, in particular for the instances with lower resolutions. This is probably because that an instance is typically smaller when it appears in first few frames. The results with 5 scales show that further adding finer-grained scales does not help much.

Anchor Scales	Small	Median	Large	Overall	mAP
2	15.9	10.2	6.6	9.9	0.545
3	13.5	9.1	6.2	8.8	0.563
4	**11.3**	**9.0**	**5.9**	**8.2**	0.562
5	11.6	9.3	6	8.4	**0.568**

Table 4. Impact of anchor scales on AD for different instance sizes. The baseline model is Faster R-CNN with ResNet-50. 2 scales: (16, 32), 3 scales: (8, 16, 32), 4 scales: (4, 8, 16, 32), and 5 scales: (4, 6, 8, 16, 32).

	VIDT				VIDT-2017
	Fold 1	Fold 2	Fold 3	Overall	
R-FCN	8.6	9.5	8.8	9.0	10.9
DFF	8.7	10.1	9.1	9.2	11.0
FGFA	9.3	11.5	10.1	10.2	12.2

Table 5. Test of significance: AD results on different sub-folds of VIDT or another different dataset demonstrate good consistency. Here DFF runs with 1 key frame out of every 2 frames.

6.4. Analysis of Variance

Given the fact that VIDT only contains a few hundreds of instances, AD of various video object detectors evaluated on this dataset might be prone to high variance. Here we analyze if our comparisons are reliable, i.e., whether the difference between AD of different methods is significant compared to variance. To test the conclusion that DFF and FGFA incur extra delay to the baseline model R-FCN, we perform a 3-fold validation to verify whether the results correlate well on each fold. In addition, we select a subset from ImageNet VID-2017 (which is recently published but not yet widely used in the community) and validate whether the same conclusion can be extended to a different dataset. The results are shown in Table 5. We find the results demonstrate good consistency across all folds and datasets.

7. Conclusion

We have presented the metric average delay (AD) to measure and compare detection delay of various video object detectors. Extensive experiments find that many detectors with descent detection accuracy suffer from the problem of increased delay. However, the widely used detection accuracy metric mAP by itself cannot reveal this deficiency. We hope our findings and the new AD metric would help the design and evaluation of future video object detectors for latency-critical tasks. We also expect large and diverse video datasets in the future and better target the delay issue.

References

[1] Berkeley DeepDrive: Low latency deep inference for self-driving vehicles. https://deepdrive.berkeley.edu/project/low-latency-deep-inference-self-driving-vehicles. 2

[2] Gedas Bertasius, Lorenzo Torresani, and Jianbo Shi. Object detection in video with spatiotemporal sampling networks. In *ECCV*, 2018. 1, 2, 6

[3] Tolga Bolukbasi, Joseph Wang, Ofer Dekel, and Venkatesh Saligrama. Adaptive neural networks for efficient inference. In *ICML*, 2017. 6

[4] Kai Chen, Jiaqi Wang, Shuo Yang, Xingcheng Zhang, Yuanjun Xiong, Chen Change Loy, and Dahua Lin. Optimizing video object detection via a scale-time lattice. In *CVPR*, 2018. 1, 5, 6

[5] Jifeng Dai, Yi Li, Kaiming He, and Jian Sun. R-FCN: Object detection via region-based fully convolutional networks. In *NeurIPS*, 2016. 1, 2

[6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *CVPR*, 2009. 1, 2, 4

[7] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (VOC) challenge. *IJCV*, 2010. 2, 5

[8] Christoph Feichtenhofer, Axel Pinz, and Andrew Zisserman. Detect to track and track to detect. In *ICCV*, 2017. 1, 2

[9] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The KITTI dataset. *International Journal of Robotics Research*, 2013. 1, 4

[10] Congrui Hetang, Hongwei Qin, Shaohui Liu, and Junjie Yan. Impression network for video object detection. *arXiv:1712.05896*, 2017. 5

[11] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. MobileNets: Efficient convolutional neural networks for mobile vision applications. *arXiv:1704.04861*, 2017. 2

[12] Kai Kang, Hongsheng Li, Junjie Yan, Xingyu Zeng, Bin Yang, Tong Xiao, Cong Zhang, Zhe Wang, Ruohui Wang, Xiaogang Wang, et al. T-CNN: Tubelets with convolutional neural networks for object detection from videos. *TCSVT*, 2018. 1, 2

[13] Yu Kong, Dmitry Kit, and Yun Fu. A discriminative model with multiple temporal scales for action prediction. In *ECCV*, 2014. 2

[14] Dong Lao and Ganesh Sundaramoorthi. Quickest moving object detection. *arXiv:1605.07369*, 2016. 2

[15] Alexander Lavin and Subutai Ahmad. Evaluating real-time anomaly detection algorithms–the numenta anomaly benchmark. In *ICMLA*, 2015. 2, 5

[16] Shih-Chieh Lin, Yunqi Zhang, Chang-Hong Hsu, Matt Skach, Md E Haque, Lingjia Tang, and Jason Mars. The architectural implications of autonomous driving: Constraints and acceleration. In *ASPLOS*, 2018. 2

[17] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *ICCV*, 2017. 2

[18] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft CoCo: Common objects in context. In *ECCV*, 2014. 1, 2

[19] Mason Liu and Menglong Zhu. Mobile video object detection with temporally-aware feature maps. In *CVPR*, 2018. 1, 2, 6

[20] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. SSD: Single shot multibox detector. In *ECCV*, 2016. 1, 2

[21] Hao Luo, Wenxuan Xie, Xinggang Wang, and Wenjun Zeng. Detect or track: Towards cost-effective video object detection/tracking. In *AAAI*, 2019. 2, 5

[22] Shugao Ma, Leonid Sigal, and Stan Sclaroff. Learning activity progression in LSTMs for activity detection and early detection. In *CVPR*, 2016. 2

[23] Behrooz Mahasseni, Xiaodong Yang, Pavlo Molchanov, and Jan Kautz. Budget-aware activity detection with a recurrent policy network. In *BMVC*, 2018. 2

[24] Huizi Mao, Taeyoung Kong, and William J Dally. CaTDet: Cascaded tracked detector for efficient object detection from video. In *SysML*, 2019. 1, 2, 3, 5, 6

[25] Huizi Mao, Song Yao, Tianqi Tang, Boxun Li, Jun Yao, and Yu Wang. Towards real-time object detection on embedded systems. *TETC*, 2018. 2

[26] Manuel Martinez, Alvaro Collet, and Siddhartha S Srinivasa. Moped: A scalable and low latency object recognition and pose estimation system. In *ICRA*, 2010. 2

[27] Sangmin Oh, Anthony Hoogs, Amitha Perera, Naresh Cuntoor, Chia-Chih Chen, Jong Taek Lee, Saurajit Mukherjee, JK Aggarwal, Hyungtae Lee, and Larry Davis. A large-scale benchmark dataset for event recognition in surveillance video. In *CVPR*, 2011. 4

[28] Vincent Poor and Olympia Hadjiliadis. Quickest detection. 2009. 2

[29] Esteban Real, Jonathon Shlens, Stefano Mazzocchi, Xin Pan, and Vincent Vanhoucke. YouTube-BoundingBoxes: A large high-precision human-annotated data set for object detection in video. In *CVPR*, 2017. 4

[30] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *NeurIPS*, 2015. 1, 2

[31] Mohammad Sadegh Aliakbarian, Fatemeh Sadat Saleh, Mathieu Salzmann, Basura Fernando, Lars Petersson, and Lars Andersson. Encouraging LSTMs to anticipate actions very early. In *ICCV*, 2017. 2

[32] Xiaodong Yang, Pavlo Molchanov, and Jan Kautz. Multilayer and multimodal fusion of deep neural networks for video classification. In *ACM Multimedia*, 2016. 2

[33] Xiaodong Yang, Pavlo Molchanov, and Jan Kautz. Making convolutional networks recurrent for visual sequence learning. In *CVPR*, 2018. 1

[34] Xitong Yang, Xiaodong Yang, Ming-Yu Liu, Fanyi Xiao, Larry Davis, and Jan Kautz. STEP: Spatio-temporal progressive learning for video action detection. In *CVPR*, 2019. 2

[35] Fisher Yu, Wenqi Xian, Yingying Chen, Fangchen Liu, Mike Liao, Vashisht Madhavan, and Trevor Darrell. BDD100K: A diverse driving video database with scalable annotation tooling. *arXiv:1805.04687*, 2018. 4

[36] Xiangyu Zhang, Xinyu Zhou, Mengxiao Lin, and Jian Sun. ShuffleNet: An extremely efficient convolutional neural network for mobile devices. In *CVPR*, 2018. 2

[37] Xizhou Zhu, Yujie Wang, Jifeng Dai, Lu Yuan, and Yichen Wei. Flow-guided feature aggregation for video object detection. In *ICCV*, 2017. 1, 6

[38] Xizhou Zhu, Yuwen Xiong, Jifeng Dai, Lu Yuan, and Yichen Wei. Deep feature flow for video recognition. In *CVPR*, 2017. 1, 2, 5, 6