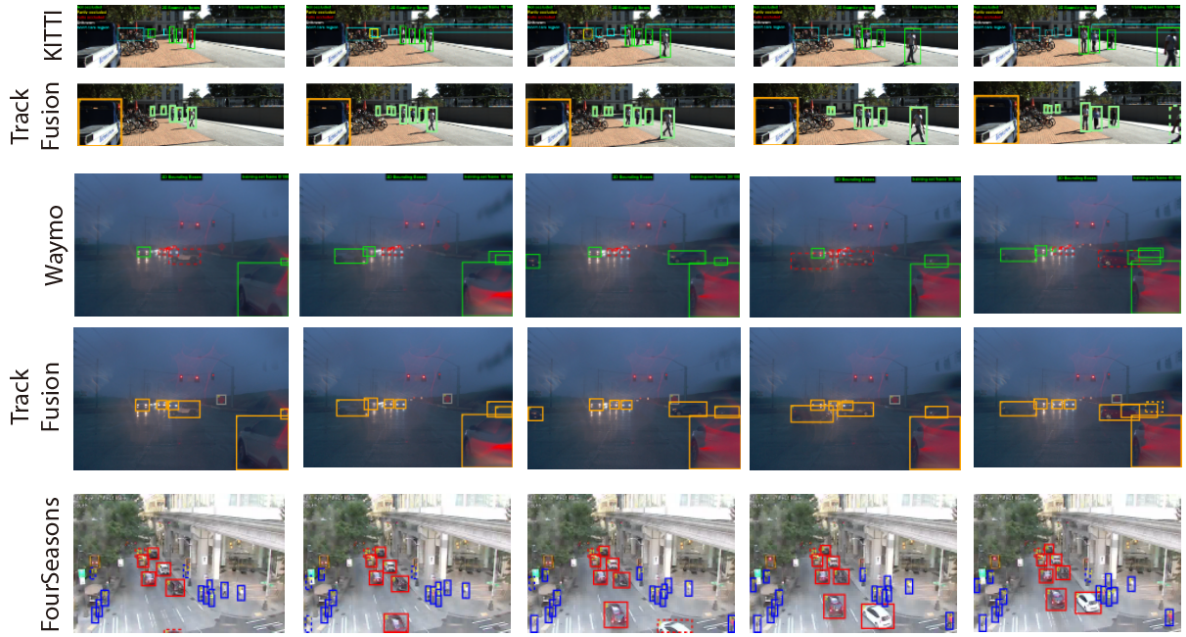
Obtaining Multi-Object Tracking Annotations at Scale

Hang Qiu
USC
hangqiu@usc.edu

Krishna Chintalapudi
Microsoft Research
krchinta@microsoft.com

Ramesh Govindan
USC
ramesh@usc.edu



Abstract

Large, accurately annotated, multi-object tracking data sets are difficult to collect. This is because real-world data can contain many objects per frame, and human annotators are unable to produce complete tracks a majority of the time. In this paper, we introduce a new fusion algorithm, TrackFusion, that produces high-quality tracks using partially correct results received from multiple annotators, while carefully weeding out poor quality results and underperforming annotators. TrackFusion is implemented on an open-source image annotation platform and exhibits over 98% precision and recall of tracks in the KITTI and Waymo data sets. We have used it to produce one of the largest available multi-object tracking data sets, FourSeasons, which has about 195,000 annotated frames spanning over a year. The figure above shows examples of TrackFusion-generated annotations on these data sets.

1. Introduction

While labeled image classification data sets have reached impressive scales [14], available data sets for multi-object tracking are much smaller. This is because accurately annotating tracks in real-world data sets is challenging for a human annotator. For example, in the KITTI [18] and Waymo [10] data sets, some frames can have up to 70 objects, and annotators can take up to a minute to annotate a single object in a 3-second video sequence (§3). As a result of this visual complexity, individual annotators are unable to produce complete annotations for the majority of tracks in those two data sets.

In this paper, we describe *TrackFusion* (§4), which combines multiple partially correct annotations from annotators to produce high quality tracks. TrackFusion continuously, and without requiring a human in the loop, estimates the quality of the fused result. It also automatically rates annotator quality, allowing better-performing annotators to be allocated work. To achieve this, TrackFusion must identify the correct parts of an annotation while rejecting the incor-

rect parts, in the face of a wide variety and range of errors that annotators can introduce into their annotations. For example, a common mistake is for annotators to miss the first few frames or the last few frames of an object. Careless annotators draw bounding boxes that are too large or too small, or those that cover the object. Annotators often miss entire object tracks, especially when there is an overwhelmingly large number of objects present in the same scene. In other cases, they draw spurious bounding boxes. In the face of these errors, TrackFusion must identify the correct parts of each annotation and then fuse them to generate high quality track annotations.

We have implemented TrackFusion on an open-source cloud-based annotation service called Satyam [27]. This service automates all aspects of annotation collection, but requires annotation developers to specify algorithms to automatically aggregate annotator contributions, and metrics to rate annotator quality. TrackFusion provides these in Satyam and is publicly available.[1]

Using Satyam, we have compared (§5) the accuracy of TrackFusion's track annotations with those provided in two publicly available data sets, KITTI [18] and Waymo [10]. TrackFusion is able to achieve over 99.8% precision and 99.1% recall on the KITTI data set and 98.4% precision and 99.1% recall on the Waymo data set. It achieves a MOTA score of 98.8 and 97.4 respectively on these data sets. These are close to the highest achievable by human annotation; these data sets use LIDAR for annotation, which can capture objects invisible to the human eye in low light conditions. Moreover, TrackFusion is able to find new tracks not annotated in those data sets often belonging to partially occluded or truncated objects.

Using TrackFusion, we have generated and made available[2] FourSeasons, one of the largest multi-object tracking data sets available today (§6). It has track annotations for cars, trucks, pedestrians, and bicycles and contains nearly 150,000 video frames (one order of magnitude larger than the KITTI and Waymo datasets). These frames span a year, so capture seasonal variability that can be used to assess temporal validity of machine learning models. Our evaluations on this data set demonstrate that models trained in one month become obsolete even for the same camera view across seasons and need to be updated.

2. Related Work

Crowd-tasking image annotations. ImageNet training data for classification is generated using Amazon Mechanical Turk (AMT), and uses majority voting for consensus [14]. Prior work [31] has used crowd-tasking for detection: high quality annotations are achieved by using workers to rate other workers, and majority voting to pick the best bounding box. Other annotation tools for more complicated machine vision tasks, such as PolygonRNN [11] for segmentation, have been proposed, but do not consider result aggregation and consensus. Third party commercial crowd-tasking systems [15, 30] and cloud-based data-labeling services [2, 4] can collect groundtruth for machine vision. Some of these support tracking annotations [2, 4], but, unlike for TrackFusion, their efficacy is not publicly known.

Crowd-tasking quality. Prior work has extensively used multiple worker annotations and majority voting to improve quality [14, 31]. For one-shot binary classification tasks, lower cost solutions exist to achieve high quality [20] or low latency [23]. For top-k classification (e.g., finding the k least blurred images in a set) several algorithms can improve crowdtasking consensus [39]. Other work has explored the cost-quality tradeoff [17] in different crowd-tasking settings: de-aliasing entity descriptions [33, 21], or determining answers to a set of questions [22]. TrackFusion is unique in focusing on multi-object tracking annotation quality.

Datasets. Several benchmark datasets exist for single object tracking [1, 6, 24, 29]. Recent multi-object tracking benchmark datasets from Waymo [10], KITTI [18], MOT [26] contain tracking annotations for vehicles and pedestrians. The FourSeasons dataset is an order of magnitude larger than these. There exist other traffic camera datasets [35, 38], some of which have tracks for vehicles [35]. FourSeasons is distinctive in being a year-long large dataset with multi-object tracks for pedestrians, cars, trucks, and pedestrians.

Tracker Designs. Tracking is a rich area in computer vision [7]. Early tracker designs separated object detection from data association [37]. Recent work has focused on improving the speed of tracking, by integrating detection and association in a single model [34], using faster Siamese networks for high-speed, yet robust association [40, 25]. TrackFusion is complementary to this line of work, and focuses on obtaining high-quality training data for evaluating these designs.

Annotation user-interfaces. Several user-interface tools permit users to annotate bounding boxes on objects in images or video frames [3, 8] and assign labels to bounding boxes. Other tools support multi-object tracking [32, 5] and segmentation [5]. TrackFusion is built on top of VATIC's user interface [32].

3. Characterizing Annotation Errors

In this section, we characterize annotator errors in multi-object tracking annotation by launching multi-object tracking tasks on Amazon Mechanical Turk (AMT) [9] using video clips from autonomous driving datasets KITTI [18] and Waymo [10]. To do this, we extended an open-source annotation crowd-sourcing platform called Satyam [27] to include multi-object tracking (§4). Our extension uses

[1] TrackFusion code: https://github.com/CoolDunes/TrackFusion
[2] FourSeasons dataset: https://trafficcamdataset.wordpress.com

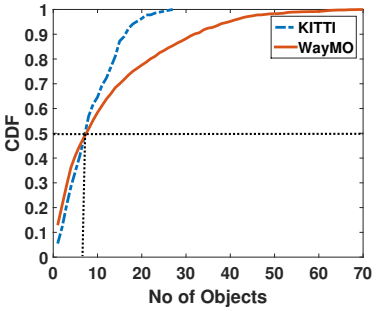


Figure 1: No. of objects in a frame

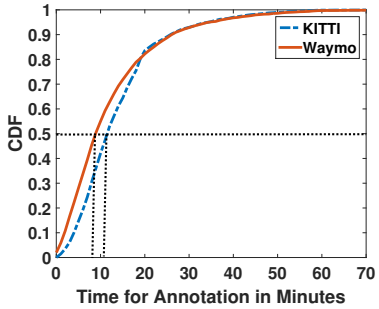


Figure 2: Time taken by workers to annotate 1 sec of video.

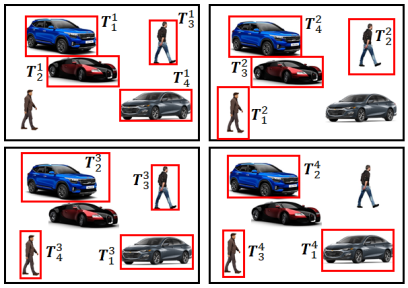


Figure 3: Track association example

VATIC [32] to annotate tracks in videos.

Experiment details. To manage the cognitive load on the annotator, we divided each video segment in the KITTI and Waymo data sets into 3-second chunks (30 video frames), resulting in 274 and 512 chunks respectively for KITTI and Waymo. We then sent each video chunk to 20 annotators on AMT, resulting in a total of 5480 and 10240 track annotations for KITTI and Waymo respectively.

The complexity of tracking annotation. In both datasets, each video frame can have a large number of objects. Figure 1 depicts the cumulative distribution of the number of objects in each frame. The median number of objects per video frame in both these data sets is 8, but the distribution has a long tail and can be as high as 70 for some frames.

In our experiments, this visual complexity resulted in long track annotation times. Figure 2 depicts the cumulative distribution of the time taken to annotate the 3-second video clip (containing a sequence of 30 frames). The median time to annotate is 10 minutes, but can be as high as 70 minutes in some cases. Thus, it takes about a minute for a worker to track a single object.

Performance of Annotators. The larger the number of objects in a video, the more error prone the annotations, since workers tend to make mistakes and even miss objects. To quantify this, we compared the annotations produced by the workers to the benchmark's annotations. We used the metrics defined in the KITTI benchmark [18] for evaluations:

- *Mostly Correct Tracks*: The annotator tracks the object correctly in > 80% of the frames.
- *Partially Correct Tracks*: The annotator tracks the object correctly in only 20-80% of the frames.
- *Mostly Incorrect/Missed Tracks*: The annotator either misses the object or tracks it correctly in less than 20% of the tracks.
- *Spurious Tracks*: The annotator draws a track where there should be none.

Table 1 lists the average values of these performance metrics across all the annotations. Annotators miss almost 43% of the tracks and only partially annotate another 11%. These experimental results point to the difficulty of obtaining high quality annotations from individual annotators and motivate the need for fusing results from multiple annotators.

	KITTI [%]	Waymo [%]
Mostly Correct Tracks	46.6	41.1
Partially Correct Tracks	13.2	9.5
Mostly Incorrect/Missed Tracks	39.3	47.1
Spurious Tracks	25	27.5

Table 1: Performance of TrackFusion on Benchmarks

4. TrackFusion

Most annotators generate about 40% of the tracks almost completely. (§3) but miss 40% of the tracks almost entirely. Assuming that while some annotators miss annotating certain tracks, others might annotate them, TrackFusion identifies and combines the partially correct parts of the annotations across multiple annotators to generate a single high quality annotation. Another significant error is that there is a significant fraction of spurious tracks (25%). These may be due to annotators accidentally drawing a bounding box and then not deleting it. Assuming that the same spurious track is not drawn by multiple workers, TrackFusion identifies tracks that are not drawn by other annotators and rejects them while generating the final annotation.

Satyam [27], on which we have implemented TrackFusion, requires new annotation developers to define two functions: a *fusion* function to combine annotator submissions, and an *annotator quality* estimator. Satyam uses the quality estimator to select annotators who have demonstrated high-quality work [27].

Given these functions, Satyam handles all other aspects of obtaining the annotations. (It requires developers to design a web user interface shown to annotators; TrackFusion embeds the VATIC [32] annotation tool in this page). In this section, we describe the fusion and annotator quality estimation functions.

Overview of TrackFusion. The functioning of TrackFusion can be described in the following steps.

1. Start by soliciting annotations from W_{min} workers.
2. *Track-Association* - identify and group tracks across annotators that correspond to the same object track.
3. *Clustering similar bounding boxes and selecting the most likely cluster* - For each frame in each track, cluster the most similar bounding boxes corresponding to the same object. Compute the likelihood for a cluster being close to the groundtruth based on annotators' historical performances. Select the maximum likelihood cluster.
4. *Accept or reject maximum likelihood clusters* - accept likelihood clusters with likelihood value greater than τ_{accept} and reject those less than τ_{reject}.
5. *Solicit more annotations* - if some maximum likelihood clusters can neither be accepted nor rejected, and the number of annotations requested does not exceed W_{max}, solicit more annotation.
6. *Combine Tracks* - If not more annotations need to be solicited, first combine the bounding boxes in each maximum likelihood cluster. Then fill in gaps *i.e*. interpolate intermediate frames where there are no annotations.
7. *Perform annotation quality assessment* - Compare each annotation to the annotation obtained by combining and compute the quality of each annotation.

We now describe TrackFusion in detail.

Notation. A multi-object tracking annotation by the m^{th} annotator $\mathbf{T}^m$ is a set of tracks $\{\mathbf{t}_1^m, \cdots, \mathbf{t}_n^m\}$, where $\mathbf{t}_i^m$ is the track corresponding to the i^{th} object drawn by the m^{th} annotator. Each track $\mathbf{t}_i^m$ is an ordered sequence of rectangular bounding boxes $< B_{i1}^m, \cdots, B_{if}^m >$ for each of the f frames in the video clip. B_ϕ is a null bounding box, indicating that no bounding box was drawn for that frame. TrackFusion takes as input a set of annotations from n annotators $\{\mathbf{T}^1, \cdots, \mathbf{T}^n\}$ to generate a fused track T^*.

Track association. Annotators may not annotate objects in the same order. Consequently, TrackFusion associates tracks that correspond to the same object into an *associated track set*. Figure 3 shows results from four annotators annotating a video with 5 objects. In this example, track association will produce the 5 associated track sets, one for each of the 5 objects *e.g*. $\{\mathbf{T}_1^1, \mathbf{T}_4^2, \mathbf{T}_2^3, \mathbf{T}_2^4\}$ corresponding to the Blue SUV in the top right, $\{\mathbf{T}_3^1, \mathbf{T}_2^2, \mathbf{T}_3^3\}$ corresponding to the person on the top right corner *etc*.

We formulate track association as a minimal weight multipartite matching on a weighted graph in which each node representing a track and there are no edges between tracks annotated by the same user. The weight of an edge between two nodes $\mathbf{t}_i^m$ and $\mathbf{t}_j^l$ measures the *similarity* between the corresponding tracks:

$$IoU_{3D}(\mathbf{t}_i^m, \mathbf{t}_j^l) = \frac{\sum_{k=1}^{k=f} \mathbf{A}\left(B_{ik}^m \cap B_{jk}^l\right)}{\sum_{k=1}^{k=f} \mathbf{A}\left(B_{ik}^m \cup B_{jk}^l\right)} \quad (1)$$

In Equation 1, $\mathbf{A}(B1 \cap B2)$ is the area of the intersection of rectangles $B1$ and $B2$ while $\mathbf{A}(B1 \cup B2)$ is the area of their union.

Multi-partitive weighted matching is NP-Hard, so we use an iterative greedy algorithm. When there are N tracks across all the annotators, the greedy algorithm starts with N sets, with one track per set. At each iteration, it merges two sets with the highest average similarity, while ensuring that each set may have at most only one track from any given annotator. The heuristic terminates when it can not merge any pair of sets. In the end, each set corresponds to all the tracks annotated by different annotators for one distinct object in the video.

Selecting bounding boxes using the compact maximum likelihood cluster. Some annotators may draw bounding boxes incorrectly (*e.g*. too large or too small) in some of the frames. To ensure that TrackFusion only includes correct annotations in the subsequent fusion, it uses a clustering algorithm to select the most likely bounding box for each object within each video frame. For clustering, TrackFusion defines the similarity of two bounding boxes B^i and B^j by the intersection-over-union: $\frac{\mathbf{A}\left(B^i \cap B^j\right)}{\mathbf{A}(B^i \cup B^j)}$.

Let $\{B^1, B^2, \cdots, B^m\}$ be the bounding boxes drawn by k different annotators for a certain video frame for the same object. Assuming that most correct bounding boxes will cluster closely, we perform hierarchical clustering on the bounding boxes. We start with m clusters, the i^{th} cluster having one element B^i. At each step, TrackFusion merges two clusters with the maximum average similarity between their elements, provided that the resulting merged cluster does not violate a *compactness constraint*. The compactness constraint ensures that no two bounding boxes within a single cluster deviate from each other (in terms of the maximum Euclidean distance between the corresponding corners of their bounding boxes) by more than a certain threshold $\tau_{compact}$ (set to 25 pixels). This threshold must be larger than the sensitivity of the input device such as a mouse [19].

At the end of the clustering, each cluster contains bounding boxes that are most similar to each other and do not deviate from each other by more than $\tau_{compact}$. TrackFusion then computes a log likelihood function for each of the clusters that measure the likelihood that the boxes are close to the groundtruth. The log likelihood function is computed as the sum of the logarithm of annotators' past success rates. The intuition is that if annotators with high success rates draw similar bounding boxes and are, therefore, part of the same cluster, then it is likely that this particular cluster of bounding boxes has good candidates that are close to the groundtruth. TrackFusion then selects the cluster with the maximum likelihood – deemed, the *maximum likelihood compact cluster* (MLCC).

Determining whether to solicit more annotations. To obtain high quality results, TrackFusion determines whether it

needs to solicit more annotations to improve the quality of a track. It starts by sending annotation requests to a minimum number of annotators W_{min} (5 in our current implementation). When it receives a new annotation, TrackFusion checks whether it needs more annotations.

To do this, it uses two thresholds on the computed likelihood of an MLCC: τ_{accept} (95% in our implementation), and τ_{reject} (20% in our implementation). If an MLCC's computed likelihood is greater than τ_{accept} (respectively, less than τ_{reject}, then TrackFusion *accepts* (respectively, *rejects*) the MLCC. The latter threshold discards spurious bounding boxes drawn by annotators, while the former ensures enough responses to obtain tighter bounding boxes.

After receiving an annotation and applying the association and clustering algorithms, if there exists an MLCC that is neither accepted nor rejected (or if it has received fewer than W_{min} annotations), TrackFusion solicits more annotations. TrackFusion also limits the maximum number of annotations to W_{max} to ensure that in some rare cases, too many annotations are not solicited. We chose $W_{max} = 20$ in our implementation.

Fusing tracks and handling gaps. Once TrackFusion determines it needs no more annotations, it fuses all the annotations into a single high track $\mathbf{T}^*$. For each MLCC, it computes a single bounding box as the weighted average of the corners of all the bounding boxes in the MLCC, using the annotator quality (described below) as the weight.

Sometimes, combining bounding boxes at the frame level might lead to discontinuities in the track, *i.e*. intermediate frames where a low likelihood or lack of annotations results in a *gap* (no bounding boxes are generated). We use linear interpolation of the corners of the bounding boxes at the beginning and end of gaps to generate bounding boxes for the intermediate frames. In addition, our annotation tool allows the annotator to explicitly mark occluded objects.

Estimating annotation quality. After generating the fused track $\mathbf{T}^*$, TrackFusion computes IoU_{3D} (Equation 1) between each submitting annotator's track $\mathbf{T}^i$ and $\mathbf{T}^*$. A track qualifies as acceptable if this value is at least 50% (the threshold used in KITTI [18]), we say the annotator *contributes to* $\mathbf{T}^*$. An annotator's quality is the fraction of accepted tracks among all the tracks annotated. This quality metric is fed back to Satyam to filter high performing annotators. Satyam only selects annotators with acceptable rate higher than 50% [27].

5. Evaluation

In this section, we evaluate TrackFusion and show that it generates high quality annotations. To do this, we compare its annotations with those provided in the KITTI [18] and Waymo [10] tracking benchmarks, using the methodology discussed in §3. We find that:

- TrackFusion matches the highest possible accuracy achievable by humans in these data sets. The benchmark data sets generate some annotations using LIDAR; especially in low light video segments, human eyes cannot discern some of these annotated objects. These account for 1-2% objects in KITTI (as confirmed by the KITTI website [18]) and between 3-5% in Waymo.
- Conversely, TrackFusion can track partially (or even completely occluded objects by interpolation). These account for about 1% of the objects. For these, both KITTI and Waymo do not provide bounding boxes, so in these cases TrackFusion generates new tracks.

	KITTI [%]	Waymo [%]
Precision per Frame	99.2	99.2
Recall per Frame	99.8	98.4
MOTA	98.8	97.4
MOTP	73.5	75.2
Mostly Correct Tracks (>80%)	99.8	94.2
Partially Correct Tracks (20-80%)	0	2.5
Missed Tracks (<20%)	0.001	3.2
Spurious Tracks	≈0.0	≈0.0
Extra Discovered Tracks	0.6	0.5

Table 2: Performance of TrackFusion on Benchmarks

Performance of TrackFusion. Table 2 presents a detailed comparison of the results, using popular metrics for multi-object tracking.

Precision and Recall per Frame: We measure precision and recall across all the video frames in the two data sets. A bounding box is correct if its IoU is greater than 50% compared to the benchmark (as recommended by KITTI [18]). In Table 2, the precision and recall for KITTI and Waymo are near-perfect: 99.2% precision and 99.8% recall for KITTI, and 99.2% precision and 98.4% recall for Waymo. As discussed above, these data sets use LIDAR to generate some annotations, and these may not be visible to humans in low light conditions. Of these, Waymo is the more challenging data set, with almost a third of the videos taken at night, which explains the slightly lower recall.

MOTA [13]: The Multi-Object Tracking Accuracy captures the aggregate of three kinds of error ratios: the ratio of misses in the sequence, the ratio of false positives, and the ratio of mismatches. As seen from Table 2, the MOTA metric values are 98.8% and 97.4% for KITTI and Waymo. These differences arise due to the use of LIDAR and the fact that TrackFusion was able track partially or even fully occluded objects using interpolation.

MOTP [13]: The Multi-Object Tracking Position is a measure of positioning accuracy and captures the IOU for matched object hypothesis pairs over all frames. Typically, values around 75% are reasonably accurate for this metric; in Table 2, the MOTP for KITTI is 73.5% and for Waymo is 75.2%. Figure 4 shows an example comparing TrackFusion

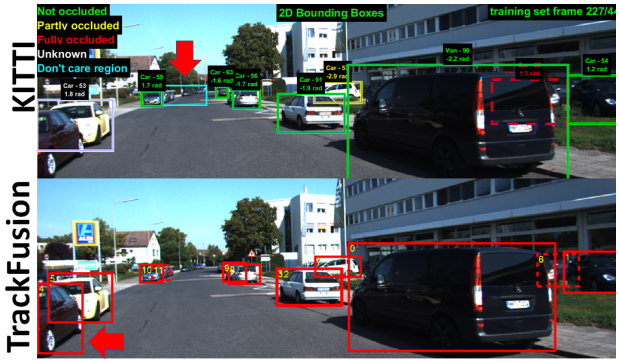


Figure 4: Example of extra tracks discovered by TrackFusion

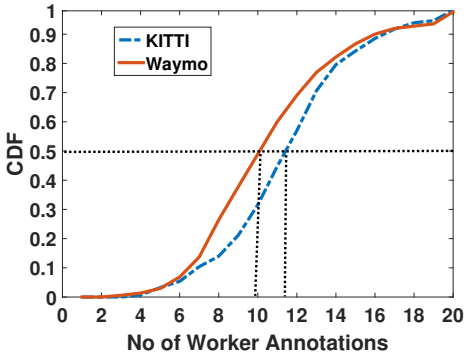


Figure 5: Distribution of number of human annotations required

and benchmark annotations. In this annotation, which are visually almost indistinguishable, the MOTP is 72.5%.

Mostly, Partially Correct Tracks and Missed Tracks. A single annotator can, on average, produce mostly correct tracks only for about 43% of tracks (§3) for the KITTI and Waymo data sets. However, with TrackFusion, these numbers increase to 99.8% and 94.2% respectively. This is because TrackFusion is able to combine the correct annotations from different annotators. Put another way, while a single annotator may produce partially correct or mostly incorrect tracks for over half the tracks in these data sets, with TrackFusion these metrics drop to near zero. Note that in case of occlusions there are gaps in the track annotations Waymo which are filled in TrackFusion. These result in mismatches between the two data sets.

Spurious Tracks. Individual annotators may generate many spurious tracks *i.e.* draw a track that should not be present (§3). TrackFusion eliminates all spurious tracks (Table 2).

Extra Discovered Tracks. TrackFusion can generate tracks for partially occluded objects for two reasons. Annotators are able to detect and annotate partial occlusions accurately. Moreover, TrackFusion interpolates bounding boxes between frames when human annotations are absent. As a consequence, TrackFusion is able to detect tracks absent in KITTI (0.6%) and Waymo (0.5%).

Incremental annotator solicitation. TrackFusion solicits more annotations when it lacks confidence in its clusters (§4). Figure 5 shows a cumulative distribution of the number of contributing annotators (§4) to tracks. Some tracks require very few (2-4) contributors, but others require 18 to 20. At the median, TrackFusion requires 12 and 11 contributing annotators per track for KITTI and Waymo respectively.

KITTI				
Maximum Annotators	5	10	15	20
Mostly Correct Tracks [%]	73.4	95.8	98.5	99.8
Partially Correct Tracks [%]	3.5	0.99	0.36	0
Missed Tracks [%]	23	3.1	1.1	0.2
Waymo				
Maximum Annotators	5	10	15	20
Mostly Correct Tracks [%]	42.9	86.1	93.6	94.2
Partially Correct Tracks [%]	14.6	7.3	2.8	2.5
Missed Tracks [%]	42.4	6.5	3.6	3.2

Table 3: Sensitivity of results to maximum number of annotators.

Sensitivity to maximum number of annotators. TrackFusion limits the maximum number of annotators per video clip; a low value of this parameter can result in too few annotations and low quality results. To understand the sensitivity of TrackFusion's performance to this parameter, Table 3 depicts the values of various tracking metrics with increasing values of this parameter. As the parameter increases from 5 to 20, the efficacy of TrackFusion's aggregation becomes apparent: the partially correct and missed tracks fall steadily, and, with a maximum of 20 annotators, TrackFusion's annotations are mostly correct 99.8% for KITTI and 94.2% for Waymo.

	With QA [%]	Without QA [%]
KITTI	68.1	86.6
Waymo	49.7	67.6

Table 4: Fraction of acceptable annotation with and without TrackFusion's QA.

The importance of annotator quality assessment. TrackFusion assesses the quality of an annotator by measuring how often he or she contributes to the final result (§4). The underlying Satyam platform uses this to recruit workers [27]. Table 4 demonstrates the efficacy of annotator quality assessment. Specifically, it shows that 20% fewer annotations are *rejected* when selecting annotators based on their quality. This reduction is significant for crowd-tasking platforms [16] and avoids wasted work.

6. FourSeasons: A New Multi-Object Tracking and Detection Data Set

Because annotating multi-object tracking is hard (§3), there exists only a few benchmark data sets for multi-object

Data Set	Pub. Frm.	Tracks	Videos	Task	Obj.	Occ.	Weather	Night	Trans-Seasonal	Pub.
MOT16 [26]	5.3k	467	7	T	P					2016
TUD [12]	610	-		D,T	P					2008
KITTI-T [18]	8k	1k	20	T	P,C	✓				2014
UA-DETRAC [36]	84k	5.9k	60	D,T	C	✓	✓	✓		2015
Waymo[3] [10]	68k	10k	73	T	B,P,C	✓	✓	✓		2019
FourSeasons	195k	56k	3839	D,T	B,P,C,T	✓	✓	✓	✓	2019

Table 5: A summary of popular publicly available multi-object tracking annotations – Pub. Frm refers to the number of published annotated frames; Tracks refers to number of object tracks; Videos to number of video segments; Tasks refers to vision tasks supported, D is detection, T is tracking; Obj refers to the types of Objects annotated, P = pedestrian, T = Truck, C = car, B= bicycle; Occ. refers to occlusions; Weather refers to weather variations; Night refers to the presence of night time video frames; Trans-seasonal refers to videos spanning multiple seasons around the year.

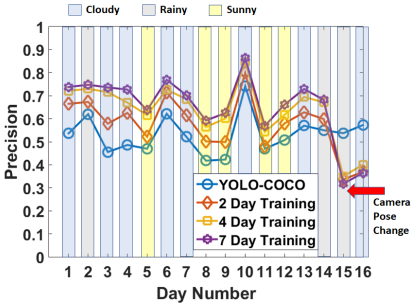


Figure 6: Effect of weather conditions

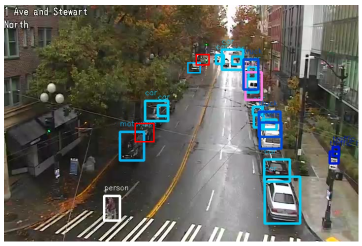


Figure 7: Fine-tuned model predictions on a *cloudy* day on CAM3. Red boxes are false negatives compared against TrackFusion annotations.

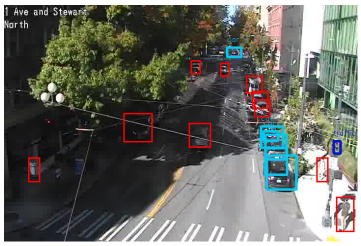


Figure 8: Long shadows as well as bright reflections results in missing detection (indicated by red boxes) on a *sunny* day on CAM3.

tracking. Table 5 depicts the characteristics of various publicly available annotated multi-object tracking data sets to the best of our knowledge. Using TrackFusion, we have generated an extensive data set, called FourSeasons, that captures the long term seasonal and diurnal variations for specific traffic surveillance cameras. Specifically, FourSeasons provides annotations from several cameras ranging from a few days to a whole year. As we show below, such data will help evaluate the generalizability of trackers to weather and seasonal effects.

As seen from Table 5, FourSeasons has a significantly larger number of publicly available annotated frames, video segments and object tracks compared to existing data sets.

6.1. Data set description

FourSeasons contains track annotations from four traffic surveillance cameras at traffic intersections in a major city in USA. We provide two distinct sets of annotations for each camera – a multi-object tracking (MOT) data set with annotated video segments and, a detection (DET) data set comprising individual images sampled from the video stream. The MOT data set was generated by recording and annotating a short video segment once every 2 hours at a frame rate of 10 frames/sec. The DET data set is generated by sampling one image each minute on a subset of the days in addition to the video frames.

Table 6 captures the time spans, duration, number of video segments, video object tracks, number of detection frames *etc*. The annotations comprise five vehicle classes: car, pedestrian, cyclist, truck, and other traffic entities (which include motorbikes, buses, trams *etc*.). Table 7 provide an insight into the distribution of various traffic entities in the MOT data set.

6.2. Case study: Effects of weather

We demonstrate the effect of weather variability on an object detector (typically used in trackers) applied to data from one of our cameras (CAM3). We use YoLoV3 [28] trained on the COCO data set to detect all cars across 16 consecutive days on CAM3. The test set has 1000 images each day. We measure the mean average precision (mAP) across all 16 days. In these days there are 5 sunny days, 8 cloudy days and 3 rainy days. As shown in Figure 6, the baseline model performs poorly with an mAP ranging between 40-60%.

A common solution is to perform transfer learning on the baseline model on data from the specific camera to improve its performance. Thus, we re-train the final fully connected layer of YoLoV3 using the first 2 days (3000 images), the first 4 days (6000 images) and the first week (10500 images) of data from CAM3. The performance of YoLoV3 improves by about 20% compared to the baseline model (Figure 6). We make two observations about these results.

Precision decreases on sunny days. In Figure 6, there is a dip in performance on the sunny days. We found that long shadows cast by cars and other objects as well as bright

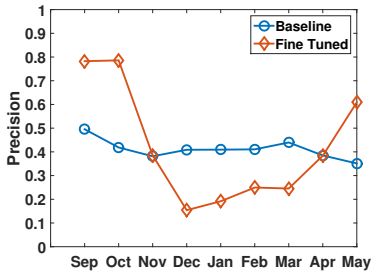


Figure 9: While a fine-tuned model performs well initially, its performance degenerates as the seasons change.

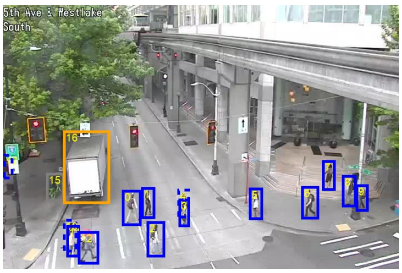


Figure 10: TrackFusion Annotations in Summer on CAM1, the tree is lush

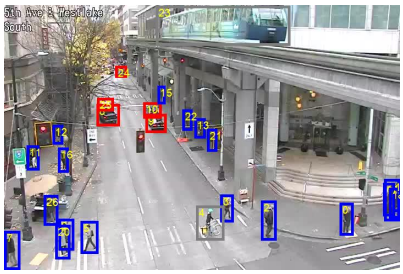


Figure 11: TrackFusion Annotations in Winter on CAM1 - Tree becomes bare causing missed detection.

Camera	Span [days]	Recording Duration	Multi-Object Tracking		Detection Frames	Resolution
			Video Segments	Object Tracks		
CAM1	393	24 Hrs	3149	44k	105k	720 X 480
CAM2	31	24 Hrs	372	6k	38k	720 X 480
CAM3	27	7A.M - 7P.M	162	3k	35k	720 X 528
CAM4	26	7A.M - 7P.M	156	3k	17k	720 X 528

Table 6: Summary of annotations across various cameras in FourSeasons MOT data set

Traffic Entity	Frames	Objects	Fraction [%]
Cars	691k	25k	56
Pedestrians	487k	18k	40
Cyclists	9k	356	0.8
Trucks	22k	780	1.8
Other Traffic Entity	29k	1022	2.3

Table 7: Distribution of various classes in CAM1

reflections from the sidewalk often confused the model. Figures 7 and 8 show an image of a cloudy and sunny day from CAM3. On the sunny day cars and pedestrians in the tree shadow as well as pedestrians on the well-lit side walk are missed by the classifier. Given the small fraction of times and hence limited training data containing days with long shadows, even transfer learning was not able to increase accuracy.

Loss of generality after transfer learning. On Day 15 and 16 the performance of the classifier that was trained on specific data from CAM3 drops dramatically even on cloudy days while that of the baseline does not. We found that on day 15, the camera's orientation was changed so that it pointed towards a different scene. While this orientation change did not affect the baseline model, all of the fine-tuned models were severely affected. This demonstrates that while transfer learning on a specific camera may improve its performance, it also makes it less general and thereby more susceptible.

6.3. Case study: Seasonal effects

We explore the effect of seasonal variations over months on the object detector. We use YoloV3 trained on the COCO data set. For each month, we sampled 1000 images randomly over the entire month to create a test set. Figure 9 depicts the performance (mAP) across various consecutive months. The model accuracy is between 40-50% throughout the year. We then performed transfer learning and fine tuned the classifier with one month of data (5000 randomly sampled images from September). While the performance of the classifier improves significantly (50% to 80%) in the months of September and October, its performance dramatically declines in the months of November to March. We found that a large tree has shed all its leaves in the scene in the Fall which causes more errors in the fine-tuned classifier (as seen in Figures 10 and 11). Performance once again begins to improve in the month of May after Spring when the trees get their leaves back. This once again demonstrates how fine-tuning a classifier can make it more sensitive and susceptible to seasonal changes in the scene.

7. Conclusions

In this paper, motivated by the difficulty of obtaining multi-object tracking annotations at scale, we have presented TrackFusion, a technique that carefully fuses the correct parts of track annotations from multiple annotators, while discarding erroneous ones. TrackFusion is implemented on the Satyam open-source annotation platform, and we have evaluated the tracks it produces on the KITTI and Waymo benchmark data sets. TrackFusion is near-perfect on those data sets, and even discovers tracks missing in those. Using TrackFusion, we have made available FourSeasons, one of the largest multi-object tracking data sets available to our knowledge. FourSeasons spans a year and can be used to understand the impact of season, weather and climate changes on tracker performance.

References

[1] A Benchmark and Simulator for UAV Tracking. https://ivul.kaust.edu.sa/Pages/Dataset-UAV123.aspx. 2

[2] Amazon sagemaker. https://aws.amazon.com/sagemaker/. 2

[3] Computer Vision Annotation Tool (CVAT). https://github.com/opencv/cvat. 2

[4] Google ai platform data labeling service. https://cloud.google.com/data-labeling/docs/. 2

[5] Playment.io Video Annotation. https://playment.io/video-annotation-tool/. 2

[6] Visual Tracker Benchmark. http://www.visual-tracking.net. 2

[7] Visual Tracking Paper List. https://github.com/foolwood/benchmark_results. 2

[8] VoTT (Visual Object Tagging Tool). https://github.com/microsoft/VoTT. 2

[9] Amazon Mechanical Turk. https://www.mturk.com/, 2018. 2

[10] Waymo open dataset: An autonomous driving dataset, 2019. 1, 2, 5, 7

[11] David Acuna, Huan Ling, Amlan Kar, and Sanja Fidler. Efficient interactive annotation of segmentation datasets with polygon-rnn++. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Jun 2018. 2

[12] M. Andriluka, S. Roth, and B. Schiele. People-tracking-by-detection and people-detection-by-tracking. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, June 2008. 7

[13] Keni Bernardin and Rainer Stiefelhagen. Evaluating multiple object tracking performance: The clear mot metrics. *J. Image Video Process.*, 2008:1:1–1:10, Jan. 2008. 5

[14] J. Deng, W. Dong, R. Socher, L. J. Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, June 2009. 1, 2

[15] Figure Eight, 2018. https://www.figure-eight.com/. 2

[16] Snehalkumar (Neil) S. Gaikwad, Durim Morina, Adam Ginzberg, Catherine A. Mullings, Shirish Goyal, Dilrukshi Gamage, Christopher Diemert, Mathias Burton, Sharon Zhou, Mark E. Whiting, Karolina R. Ziulkoski, Alipta Ballav, Aaron Gilbee, Senadhipathige S. Niranga, Vibhor Sehgal, Jasmine Lin, Leonardy Kristianto, Angela Richmond-Fuller, Jeff Regino, Nalin Chhibber, Dinesh Majeti, Sachin Sharma, Kamila Mananova, Dinesh Dhakal, William Dai, Victoria Purynova, Samarth Sandeep, Varshine Chandrakanthan, Tejas Sarma, Sekandar Matin, Ahmed Nasser, Rohit Nistala, Alexander Stolzoff, Kristy Milland, Vinayak Mathur, Rajan Vaish, and Michael S. Bernstein. Boomerang: Rebounding the consequences of reputation feedback on crowdsourcing platforms. *CoRR*, abs/1904.06722, 2019. 6

[17] H. Garcia-Molina, M. Joglekar, A. Marcus, A. Parameswaran, and V. Verroios. Challenges in data crowdsourcing. *IEEE Transactions on Knowledge & Data Engineering*, 28(4):901–911, April 2016. 2

[18] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012. 1, 2, 3, 5, 7

[19] C. Hoffmann. Mouse DPI and Polling Rates Explained. https://www.howtogeek.com/182702/mouse-dpi-and-polling-rates-explained-do-they-matter-for-gaming/. 4

[20] David R. Karger, Sewoong Oh, and Devavrat Shah. Iterative learning for reliable crowdsourcing systems. In J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 24*, pages 1953–1961. Curran Associates, Inc., 2011. 2

[21] Asif R. Khan and Hector Garcia-Molina. Attribute-based crowd entity resolution. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, CIKM '16, pages 549–558, New York, NY, USA, 2016. ACM. 2

[22] Asif R. Khan and Hector Garcia-Molina. Crowddqs: Dynamic question selection in crowdsourcing systems. In *Proceedings of the 2017 ACM International Conference on Management of Data*, SIGMOD '17, pages 1447–1462, New York, NY, USA, 2017. ACM. 2

[23] Ranjay Krishna, Kenji Hata, Stephanie Chen, Joshua Kravitz, David A. Shamma, Fei-Fei Li, and Michael S. Bernstein. Embracing error to enable rapid crowdsourcing. *CoRR*, abs/1602.04506, 2016. 2

[24] Matej Kristan, Jiri Matas, Aleš Leonardis, Tomas Vojir, Roman Pflugfelder, Gustavo Fernandez, Georg Nebehay, Fatih Porikli, and Luka Čehovin. A novel performance evaluation methodology for single-target trackers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(11):2137–2155, Nov 2016. 2

[25] B. Li, J. Yan, W. Wu, Z. Zhu, and X. Hu. High performance visual tracking with siamese region proposal network. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8971–8980, June 2018. 2

[26] Anton Milan, Laura Leal-Taixe, Ian Reid, Stefan Roth, and Konrad Schindler. Mot16: A benchmark for multi-object tracking, 2016. 2, 7

[27] Hang Qiu, Krishna Chintalapudi, and Ramesh Govindan. Satyam: Democratizing groundtruth for machine vision. *CoRR*, abs/1811.03621, 2018. 2, 3, 5, 6

[28] Joseph Redmon, Santosh Kumar Divvala, Ross B. Girshick, and Ali Farhadi. You Only Look Once: Unified, Real-Time Object Detection. *CoRR*, abs/1506.02640, 2015. 7

[29] A. W. M. Smeulders, D. M. Chu, R. Cucchiara, S. Calderara, A. Dehghan, and M. Shah. Visual tracking: An experimental survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(7):1442–1468, July 2014. 2

[30] Spare5 Website. https://app.spare5.com/fives, 2018. 2

[31] Hao Su, Jia Deng, and Li Fei-Fei. Crowdsourcing annotations for visual object detection. In *The 4th Human Computation Workshop, HCOMP@AAAI 2012, Toronto, Ontario, Canada, July 23, 2012.*, 2012. 2

[32] Vatic: Video annotation tool from irvine, ca, 2018. https://github.com/cvondrick/vatic. 2, 3

[33] Vasilis Verroios, Hector Garcia-Molina, and Yannis Papakonstantinou. Waldo: An adaptive human interface for crowd

entity resolution. In *Proceedings of the 2017 ACM International Conference on Management of Data*, SIGMOD '17, pages 1133–1148, New York, NY, USA, 2017. ACM. 2

[34] Zhongdao Wang, Liang Zheng, Yixuan Liu, and Shengjin Wang. Towards real-time multi-object tracking, 2019. 2

[35] Longyin Wen, Dawei Du, Zhaowei Cai, Zhen Lei, Ming-Ching Chang, Honggang Qi, Jongwoo Lim, Ming-Hsuan Yang, and Siwei Lyu. DETRAC: A new benchmark and protocol for multi-object tracking. *CoRR*, abs/1511.04136, 2015. 2

[36] Longyin Wen, Dawei Du, Zhaowei Cai, Zhen Lei, Ming-Ching Chang, Honggang Qi, Jongwoo Lim, Ming-Hsuan Yang, and Siwei Lyu. Ua-detrac: A new benchmark and protocol for multi-object detection and tracking, 2015. 7

[37] Nicolai Wojke, Alex Bewley, and Dietrich Paulus. Simple online and realtime tracking with a deep association metric. *CoRR*, abs/1703.07402, 2017. 2

[38] Shanghang Zhang, Guanhang Wu, João Paulo Costeira, and José M. F. Moura. Understanding traffic density from large-scale web camera data. *CoRR*, abs/1703.05868, 2017. 2

[39] Xiaohang Zhang, Guoliang Li, and Jianhua Feng. Crowdsourced top-k algorithms: An experimental evaluation. *Proc. VLDB Endow.*, 9(8):612–623, Apr. 2016. 2

[40] Zheng Zhu, Qiang Wang, Bo Li, Wei Wu, Junjie Yan, and Weiming Hu. Distractor-aware siamese networks for visual object tracking. *CoRR*, abs/1808.06048, 2018. 2