TrackVerse: A Large-scale Dataset of Object Tracks for Visual Representation Learning

Supplementary Material

A. Dataset Analysis and Examples

Fig. 1 provides further analysis of the dataset. Fig. 1a illustrates the annotation density of TrackVerse by showing the number of tracks per unit of time against the total duration in seconds. While the majority of the dataset comprises time units with fewer tracks, the graph also highlights instances of highly complex scenes where more than 26 objects are detected per unit time. Fig. 1b shows the motion statistics of all object tracks in the dataset. The motion intensity is calculated as the 90th quantile of the optical flow magnitudes within a frame and averaged over time. Fig. 1c further breaks down the motion intensity among the 50 most common classes. As expected, the motion intensity varies significantly across different object classes, with static objects like *coffee table* and *sofa* presenting the lowest motion intensity, and objects often depicted in motion like *baseball cap* and *dog* the highest. Fig. 1d shows the redundancy statistics between tracks of a given class. To measure redundancy we sample 50 tracks per class and calculate the nearest neighbor distance between their ResNet-50 features. Although we have identified some examples with high redundancy, where the same object is tracked multiple times, the majority of classes have low redundancy, indicating that the dataset is diverse and contains a wide variety of object instances.

Failure Modes There are several ways in which the pipeline can produce suboptimal object tracks. Shown in Fig. 3 are examples of the main types of possible failures. Although the failure modes are difficult to quantify within our dataset, they are relatively rare and we believe they do not significantly affect the overall quality of the representations learned in the dataset. However, they are important to consider when using the dataset for training and evaluation.

- Object Switch: This occurs when one track focuses on different objects at different points in time. This is the worst type of failure as it is detrimental to the variance-aware contrastive framework, i.e., two different objects would be trained to be predictably variant from one another. This type of failure appears to be rare as the tracking method heavily weights appearance features.
- Cartoon: Although we filtered out cartoons before detection and tracking, not all were correctly identified.
- Incorrect Label: Some tracks have incorrect labels, which disrupts the class balancing of the dataset. Typically, incorrectly labeled tracks still share some attributes with the predicted class (in the example in Fig. 3 the guitar body has a similar color and glossy appearance to that of a piggy bank).
- ID Switch: This is the opposite case as Object Switch. One object fails to be a single contiguous track and instead belongs to several tracks. This may happen due to quick changes in the scene or long occlusions.

B. Pipeline Implementation Details

B.1. Scene Segmentation

To split videos into individual scenes, we employ the AdaptiveDetector algorithm provided by PySceneDetect [2] Python API. The AdaptiveDetector functions as a two-pass system: it begins by employing the ContentDetector to evaluate changes in frame content, identifying fast cuts by analyzing shifts in color and intensity between consecutive frames. Specifically, changes are measured in the HSV color space and compared against a set threshold to pinpoint the occurrence of a fast cut, indicating a scene transition. In its second pass, the AdaptiveDetector applies a rolling average to the initial frame scores. This method helps mitigate false detections that can arise from camera movements and other similar disturbances, thereby enhancing the accuracy of the scene segmentation.

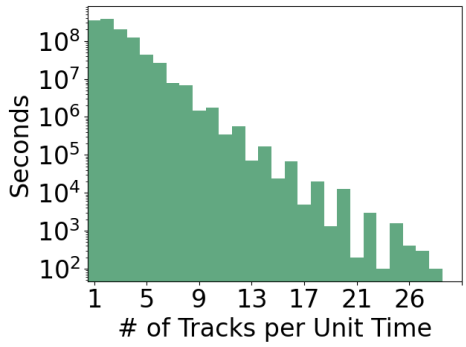
B.2. Scene Filtering

Two content-aware filters are used to remove unwanted scenes from the dataset.

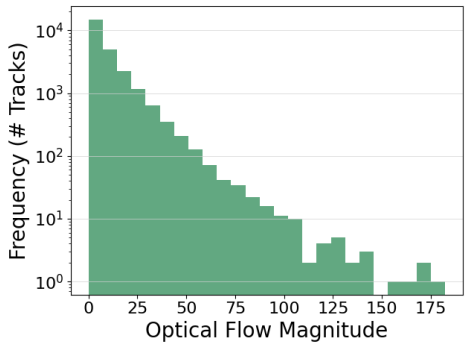
Cartoon Filter After downloading the initial set of videos, we encountered an abnormally large number of cartoon content (over 20%). Since our goal is to learn representations of natural objects, we trained and deployed a cartoon detector for filtering. The model (an ImageNet-pretrained ResNet18) was finetuned on Cartoon classification dataset[1] containing 100k cartoon images and real images randomly sampled from TrackVerse. The model was trained for 10 epochs with a batch size of 4 and a learning rate of 0.0001. We used this model to remove all segments predicted as cartoons with a confidence threshold of 0.5.

Static Visual Content Filter We also filtered out scenes with static visual content using a static visual filter. This filter identifies and removes videos with minimal frame-to-frame changes, ensuring that dynamic and informative content is retained. The filter processes each video frame by frame, comparing consecutive grayscale frames to calculate the rate of pixel change. Videos where the average change

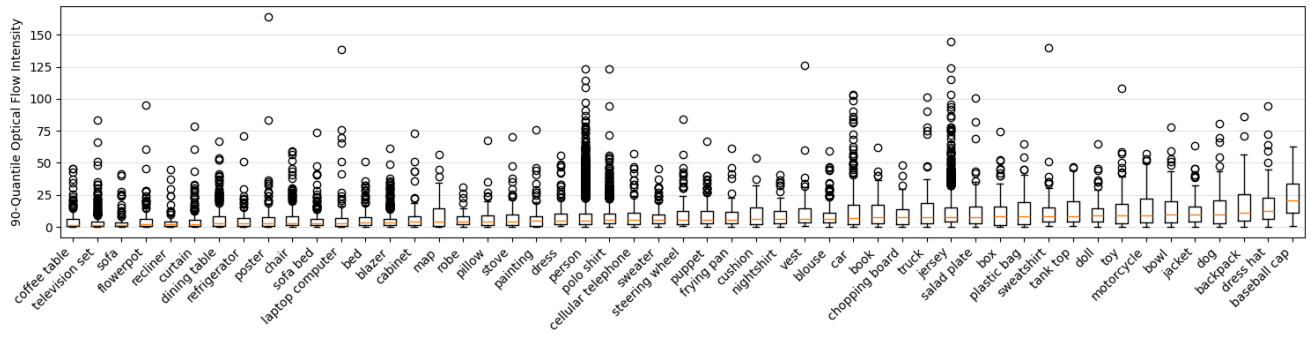
[1] https://www.kaggle.com/datasets/volkandl/cartoon-classification



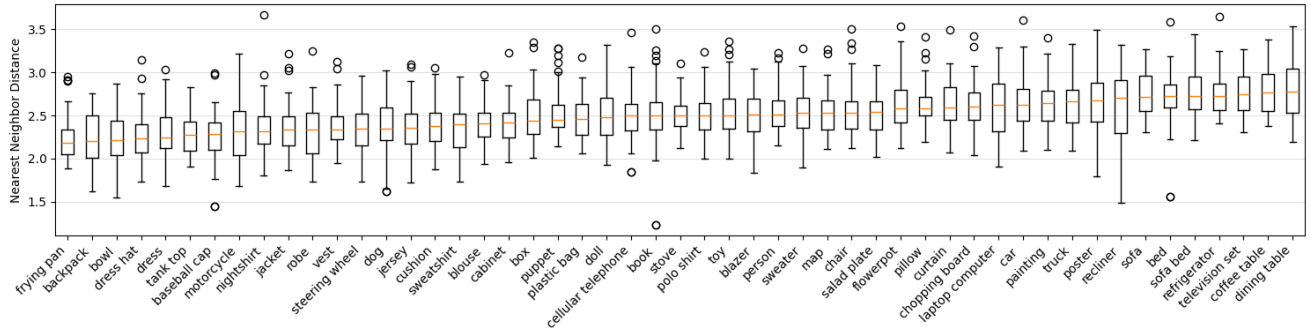
(a) Annotation Density



(b) Motion



(c) Motion intensity among the 50 most common predicted classes.



(d) Within class track redundancy among the 50 most common predicted classes.

Figure 1. **Additional analysis on the Full `TrackVerse`.** (a) The total cumulative duration in seconds that had a specific number of tracks per unit time; (b) Motion statistics of all object tracks; (c) Box plot of optical flow magnitudes (90th quantile magnitude within a track) across the 50 most common object categories; (d) Box plot of nearest neighbor distances between ResNet-50 features of tracks within a class across the 50 most common object categories.

rate remains below a threshold of 0.1 are classified as static and excluded from the dataset. This approach helps focus the dataset on visually dynamic content suitable for representation learning.

B.3. Object Parsing

When optimizing the object tracking pipeline, both accuracy and efficiency play a crucial role in creating a large-scale dataset of object tracks. Decreasing the size of input images to DETIC is one of the most effective ways to reduce detection time, but this also degrades performance. To evaluate this tradeoff, we evaluated the performance of DETIC on the LVIS validation data for a variety of input sizes. The results of these tests can be found in Fig. 4. Fig. 4a shows the miss rates at different frame sizes; an object is considered missed if there are no high confidence detections (defined as confidence > 0.55) with an IOU greater than 0.5. Classification accuracy (Fig. 4b) is reported for all detected objects. Fig. 4c shows the average time detection takes per image. Choosing an input frame size of 480 balances low miss rate/high accuracy with low detection time. For all detected objects, DETIC has a top-1 accuracy of 69.3% and a top-5 accuracy of 92.0%.

Several parameters of the detection-tracking pipeline are

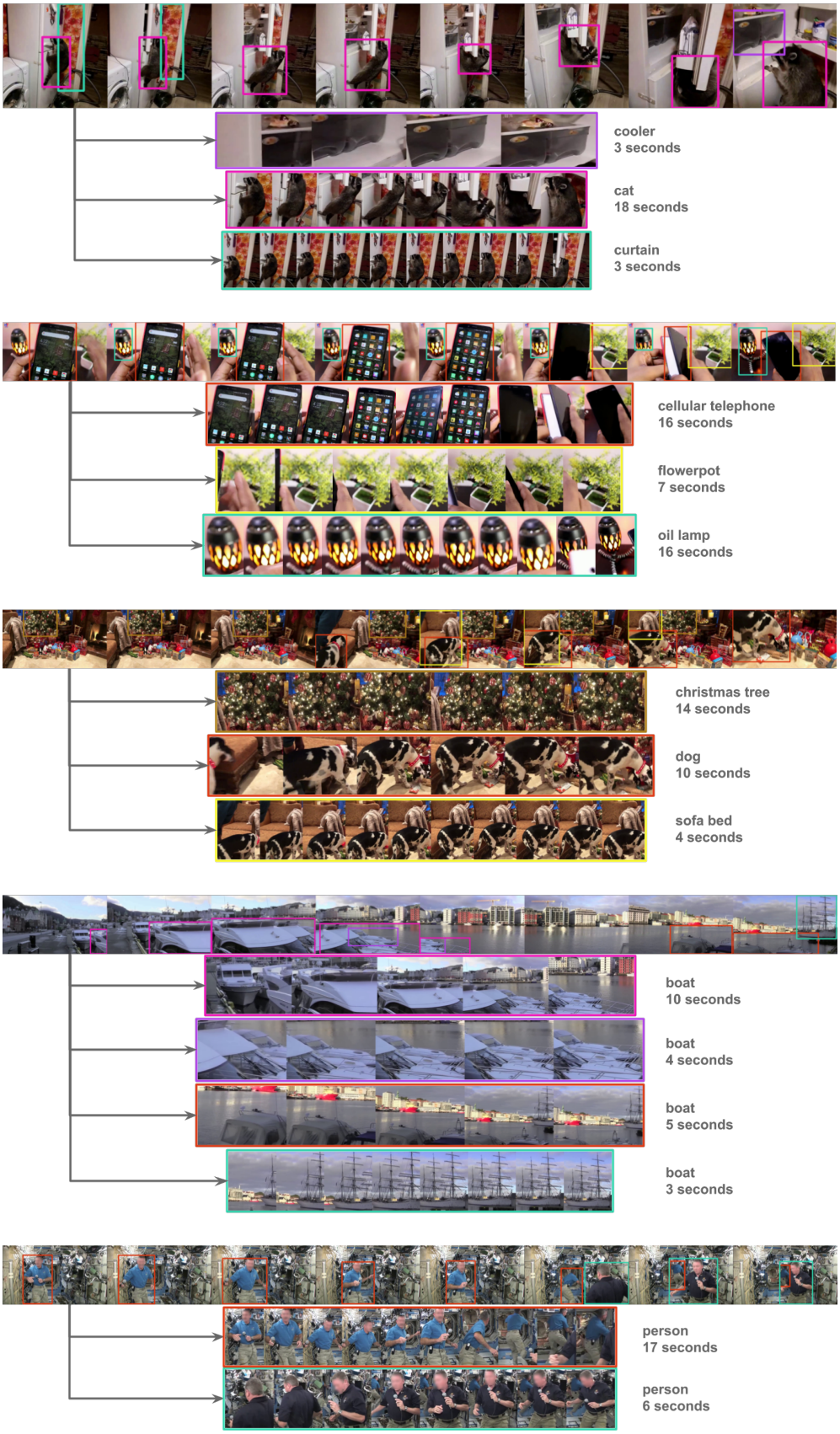


Figure 2. **Dataset example.** From each video clip, all objects are localized and tracked over time to form the TrackVerse dataset.

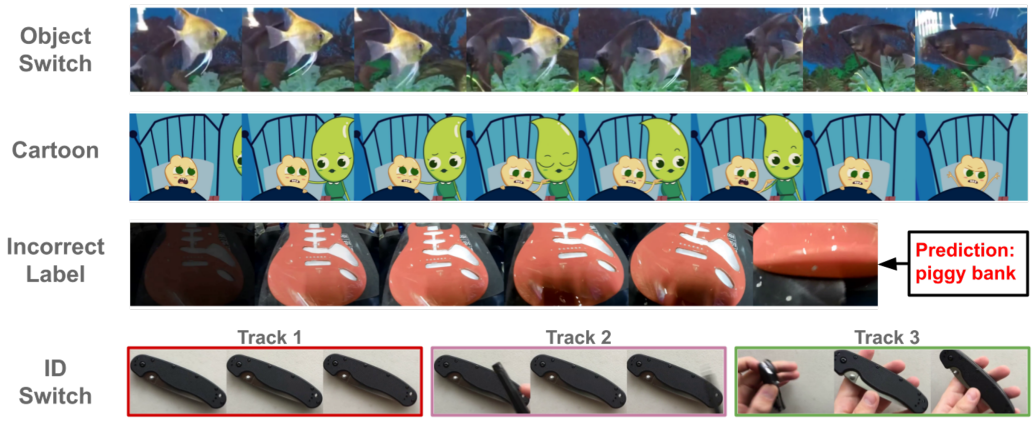
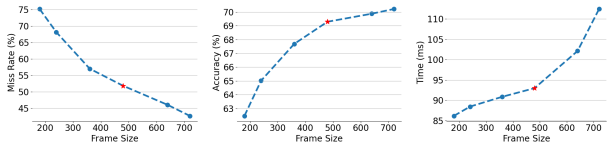


Figure 3. Failure cases within the dataset.



(a) Miss Rate (b) Accuracy (c) Detection Time

Figure 4. Frame size evaluation on the LVIS dataset. The chosen configuration is highlighted in red.

critical for tracking quality. The 2016 Davis dataset [7] is a video object segmentation dataset. This dataset was chosen for evaluating our pipeline as the segmentations are a variety of human, animal, and inanimate objects. All segmentation masks were converted to bounding boxes for evaluation purposes.

We used the Davis dataset to optimize 8 different parameters of the tracking pipeline. Specifically,

- Confidence Threshold: All detections with confidences below this threshold are removed.
- Frame Rate: Frames per second, all videos are originally 30 fps.
- Frame Size
- Matching Threshold: A high confidence detection matches to a track if the matching score is above this threshold.
- Motion Weight: Amount to weight motion information (i.e., bounding box overlap) compared to appearance information (i.e., Detic features) when matching detections to tracks. Appearance weight and motion weight always sum to 1.
- Non-maximum suppression (NMS): Removes all but the highest confidence detection for all detections with IOU overlap greater than this threshold.
- IOU Low Threshold: A low confidence detection matches a track if its IOU with the track is above this threshold.
- Tracking Threshold: A new track is created if a detection's confidence is above this threshold and does not match any existing tracks.

For evaluation, we primarily use Multiple Object Tracking Accuracy (MOTA), defined as

$$MOTA = 1 - \frac{FN + FP + IDS}{T}, \quad (1)$$

where FN is the number of false negatives (i.e., missed detections), FP is the number of false positives (i.e., additional detections not registered by Davis), IDS is the number of ID switches (i.e., objects detected by multiple tracks), and T is the total number of ground truth bounding boxes. Since the Davis dataset only has annotations for the primary subjects of each video, our pipeline can track correct objects that do not have corresponding ground truth labels. Due to this, minimizing false negatives was considered more important than minimizing false positives. In any case, where MOTA was relatively consistent between parameters, the total number of false negatives was used to choose parameters. Parameter-specific tests for MOTA scores and the number of false negatives can be seen in Fig. 5 and Fig. 6, respectively.

C. Model and Training Configuration

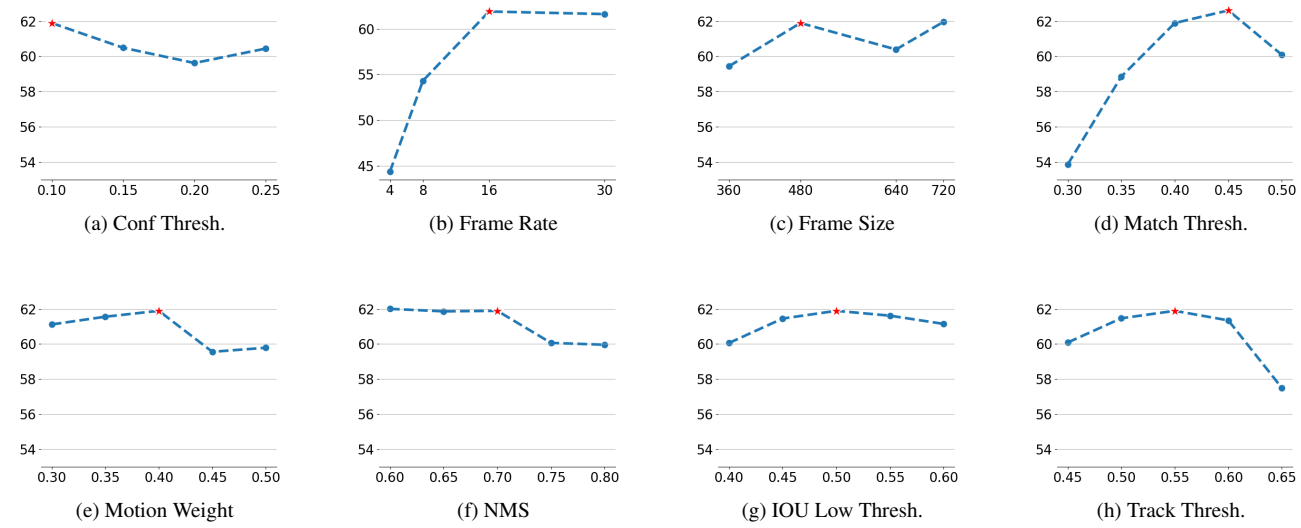
To ensure reproducibility, we provide detailed implementation details for all our experiments in Section 5 of the main text.

C.1. Pretraining

Tab. 1 shows the hyperparameters used for pre-training. We adopt 8 RTX A4500 GPUs with an effective 1024 batch size (128 per GPU×8 GPUs).

C.2. TrackVerse Few-shot Learning

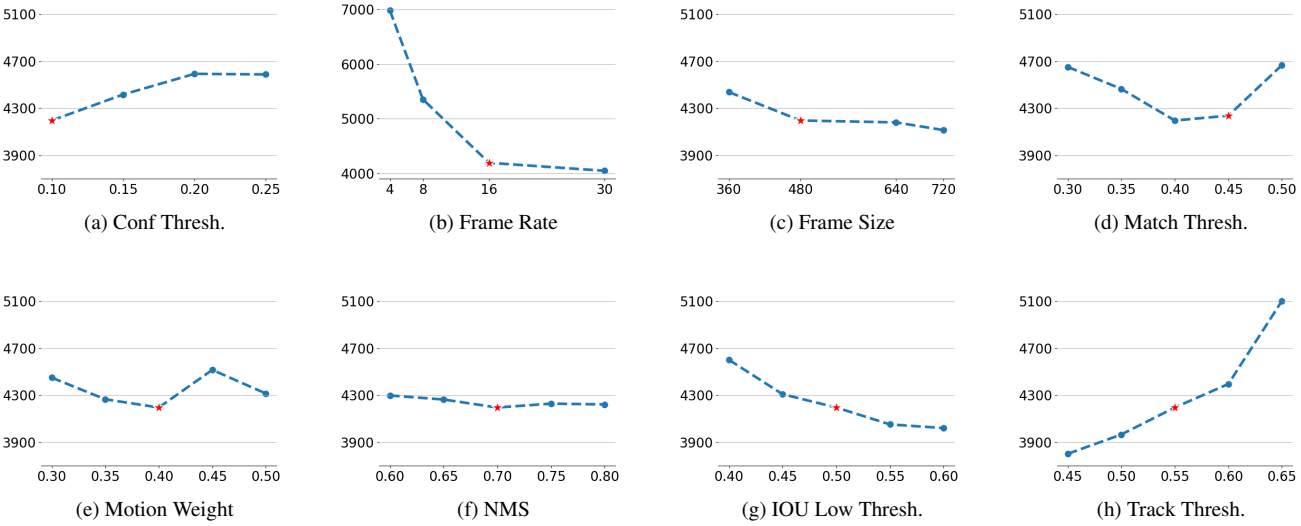
We conduct few-shot learning on the TrackVerse validation set, which contains 505 categories and 6 images per category. We run the evaluation with the same configuration



(a) Conf Thresh. (b) Frame Rate (c) Frame Size (d) Match Thresh.

(e) Motion Weight (f) NMS (g) IOU Low Thresh. (h) Track Thresh.

Figure 5. MOTA scores on Davis while varying parameter configurations. The chosen configuration is highlighted in red.



(a) Conf Thresh. (b) Frame Rate (c) Frame Size (d) Match Thresh.

(e) Motion Weight (f) NMS (g) IOU Low Thresh. (h) Track Thresh.

Figure 6. Number of False Negatives on Davis while varying parameter configurations. The chosen configuration is highlighted in red.

(Tab. 2) on 6 different sets of 5 training images per class and average the final performance.

C.3. Action Recognition on SSv2

We fine-tuned the TimeSformer model, initializing the spatial attention block with weights from the pre-trained model and the temporal attention block with zeros to utilize the learned spatial representation, as shown in Fig. 7. The default configuration details are provided in Tab. 3. All models were finetuned for 20 epochs using 8 frames per video, SGD optimizer, weight decay of $1e^{-4}$, batch size of 16, gradient accumulation set to 2, and a cosine decay learning rate schedule. The base learning rate was set to 1.0 with 4 warm-up epochs. To optimize the training settings, we conducted a parameter sweep covering learning rates of 0.1, 0.5, 1.0, and 3.0; weight decay options of 0, 0.0001, and 0.001; and fine-tuning epochs of 10, 15, 20, and 30.

C.4. Object-Attribute Classification on MIT-States

We fine-tuned a plain CLIP model following [9], where the image encoder was initialized with our pre-trained model, as shown in Fig. 8. The default configuration details are provided in Tab. 4. We conducted a parameter sweep over weight decay of 0, 0.005, 0.05, and 0.1, drop path rate of 0, 0.1, and 0.2 to find the optimal setting.

Table 1. Pretraining config.

Config	Value
Backbone	ViT-B/16
Dataset	TRACKVERSE 1121K-CB2500
Optimizer	AdamW
Warmup Updates	10,000
Epochs	209
Batch size	1024
LR Schedule	Cosine Decay w/ Warmup
Base LR	1.5e-4
Weight Decay	0.2
Gradient Clip	0.5
Optimizer Momentum	$\beta_1 = 0.9$, $\beta_2 = 0.95$
Target Encoder Momentum	$0.99 \rightarrow 1.0$
Spatial Augmentations	RandomHorizontalFlip RandomResizedCrop(224,(0.2,1))
Color Augmentations	RandomColorJitter($p = 0.8$) RandomGrayscale($p = 0.2$) RandomAdaptiveGaussianBlur((0,1.)) RandomSolarize($p = 0.1$,)
Temporal Augmentations	$\delta = 2$

Table 2. TRACKVERSE few-shot learning config.

Config	Value
Dataset	TRACKVERSE Val
Optimizer	SGD
Epochs	30
Batch size	128
LR Schedule	Cosine Decay w/ Warmup
Base LR	0.03
Warmup Epochs	5
Weight Decay	0
Optimizer Momentum	0.9
Data Augmentations	RandomHorizontalFlip RandomResizedCrop

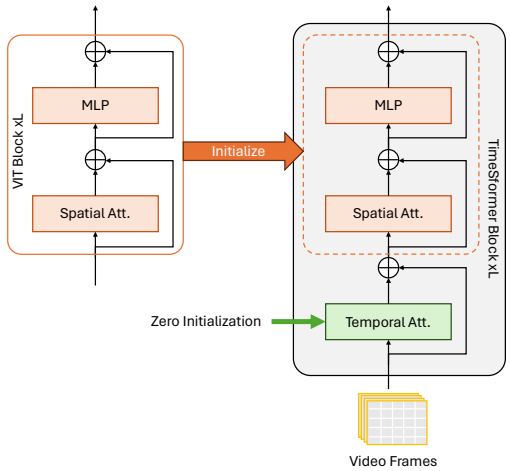


Figure 7. **Fine-Tuning TimeSformer with TrackVerse Pretrained Features on SSv2.** The spatial attention block (highlighted in orange) is initialized with weights from the pre-trained model, while the temporal attention block is initialized with zeros.

Table 3. **SSv2 Ft config.**

Config	Value
Dataset	SSv2 [5]
Optimizer	SGD
Epochs	20
Batch size	16
LR Schedule	Cosine Decay w/ Warmup
Base LR	1.0
Warmup Epochs	4
Weight Decay	0.0001
Layer Decay	0.8
Data Augmentations	RandAug(7,4,0.5,1)
Label Smoothing	0.0
Random pixel erase	0.25
Mix-Up	0.0
Cut-Mix	0.0
Path Drop	0.1
Attn Drop	0.0

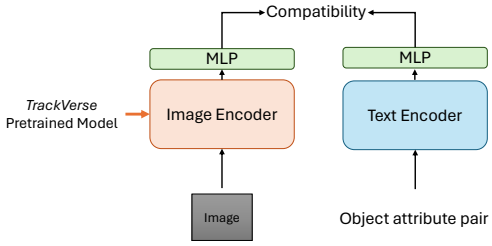


Figure 8. **Fine-Tuning CLIP with TrackVerse Pretrained Features.** The image encoder is initialized with weights from the TRACKVERSE-pretrained model to leverage learned image features. The text encoder processes object-attribute pairs, and both encoders are connected to an MLP layer to compute compatibility between visual and textual representations.

Table 4. **MIT-States Ft config.**

Config	Value
Dataset	MIT States [6]
Optimizer	AdamW
Epochs	40
Batch size	256
LR Schedule	Cosine Decay w/ Warmup
Base LR	0.000015
Warmup Epochs	20
Weight Decay	0.01
Optimizer Momentum	$\beta_1 = 0.9$, $\beta_2 = 0.999$
Data Augmentations	RandomHorizontalFlip RandomResizedCrop

D. Additional SSL Method Ablation

D.1. Variance-Aware Contrastive Learning

We analysis the impact of various components on the performance of the variance-aware contrastive learning through an ablation study. As outlined in the main text, the most effective configuration involves running variance-aware MoCo on a class-balanced subset and conditioning the predictor on the spatiotemporal metadata of each crop. The inclusion of Adaptive Gaussian blur (with a maximum blurriness of 1) and temporal augmentations (featuring temporal jittering and a δ value of 2) also significantly enhances

Table 5. **Ablation studies around the optimal congiguration**. We pre-train on the 184K-CB300 subset for 125 epochs. We report nearest neighbor (NN) accuracy (%) on LVIS-ImageNet (LVIS-IN) set, NN and few-shot learning (FSL) accuracy (%) on TrackVerse validation set, and NN ccuracy (%) on MIT-States object classification (Obj.), attribute classification (Attr.), and object-attribute pair classification (Pair) tasks. The default configuration is marked in blue.

(a) Class Distribution

Subset	LVIS-IN	TrackVerse		MIT-States		
	NN	NN	FSL	Obj. NN	Attr. NN	Pair NN
184K-Random	26.0	22.9	32.5	46.3	34.3	8.3
184K-CB300	**27.7**	**26.6**	**36.9**	**48.3**	**34.4**	**8.7**

(b) Temporal Augmentation

Temporal Jitter	Time Gap δ	LVIS-IN	TrackVerse		MIT-States		
		NN	NN	FSL	Obj. NN	Attr. NN	Pair NN
✗	0	23.3	19.3	28.4	43.5	32.0	7.4
✓	0	23.8	19.5	29.4	43.7	32.1	7.5
✓	2	**27.7**	**26.6**	**36.9**	**48.3**	**34.4**	**8.7**
✓	5	25.7	24.1	34.3	46.5	34.1	8.2

(c) Adaptive Gaussian Blur

Adaptive Gaussian Blur	Max Blur	LVIS-IN	TrackVerse		MIT-States		
		NN	NN	FSL	Obj. NN	Attr. NN	Pair NN
✗	2	27.5	26.1	35.0	47.4	34.3	8.6
✗	0	27.6	26.3	36.5	**48.3**	**34.6**	**8.7**
✓	1	**27.7**	**26.6**	**36.9**	**48.3**	34.4	**8.7**
✓	2	27.5	26.6	36.6	48.2	34.1	8.6

(d) View Condtion

View Condition	LVIS-IN	TrackVerse		MIT-States		
	NN	NN	FSL	Obj. NN	Attr. NN	Pair NN
✗	26.4	24.6	32.7	46.8	34.8	8.4
Spatial	27.4	25.1	35.4	48.2	33.9	8.6
Temporal	27.6	**26.6**	35.7	47.2	**34.9**	**8.9**
Spatial+Temporal	**27.7**	**26.6**	**36.9**	**48.3**	34.4	8.7

the model's performance. To further understand the contribution of each component, we conduct pre-training on the TrackVerse 184k-cb300 subset both with and without each element of the optimal configuration. The results of this evaluation are presented in Tab. 5.

Class Distribution Although the full TrackVerse contains a diverse range of object classes, the class distribution is highly imbalanced. Given a fixed dataset size, class imbalance significantly reduces the dataset diversity, leading to suboptimal model performance. This effect is once again highlighted in Tab. 5a, where pretraining on a class-balanced subset consistently outperforms an equally sized randomly selected subset across all tasks.

Temporal Augmentation As demonstrated in Tab. 5b, the introduction of temporal jittering and a time gap can enhance the model's performance. While a certain degree of time gap between two views is beneficial for capturing temporal variations, an excessively large gap can lead to significant differences in object appearance and increased uncertainty. Also, since there is a large number of small tracks, increasing the time gap results in less temporal jittering, which can be detrimental to the model's performance.

Adaptive Gaussian Blur The Random Gaussian Blur augmentation applies a blur to each image, with the level of blurriness controlled by $\sigma \sim \mathcal{U}(\sigma_{\text{min}}, \sigma_{\text{max}})$. Typically, $\sigma_{\text{min}} = 0$ and $\sigma_{\text{max}} = 2$ [3]. However, since the TrackVerse dataset contains tracks of various sizes, smaller tracks will naturally be depicted with less detail when upscaled to the model's input resolution of 224. Therefore, we introduce the Adaptive Gaussian Blur augmentation. This method dynamically adjusts the level of Gaussian blur according to the subsampling factor of each track. The subsampling factor is calculated as

$$s = \sqrt{\frac{w_{\text{track}} \times h_{\text{track}}}{w_{\text{video}} \times h_{\text{video}}}}, \tag{2}$$

The Adaptive Gaussian Blur samples the adjusted Gaussian kernel as $\sigma = \frac{1.5}{s} \times \mathcal{U}[0, \sigma_{\text{max}}]$, where the coefficient 1.5 is the average factor across the dataset.

Tab. 5c compares the effects of random Gaussian Blur, no blur, and the adaptive Gaussian Blur with varying σ_{high} values. Interestingly, omitting Gaussian Blur yields better performance than the basic random Gaussian Blur. However, the Adaptive Gaussian Blur, when applied with a less intense blurriness level, delivers the optimal performance. These findings suggest that video dataset pretraining benefits from a milder Gaussian Blur compared to static image training. Excessive blurring tends to obscure critical details necessary for effective learning, especially when the original videos already contain motion blur. Therefore, adjusting the blurriness level to a lower intensity aligns better with the intrinsic properties of the TrackVerse, thereby enhancing the overall model effectiveness.

View Condition As illustrated in Tab. 5d, the application of variance-aware prediction with even a single source of view condition can significantly enhance model performance across all tasks, outperforming the traditional view-invariance approach. The independent utilization of either spatial or temporal conditions can provide a substantial performance boost, with the temporal condition proving to be more effective than the spatial condition alone. However, the combination of both spatial and temporal conditions yields the most optimal performance in the majority of tasks.

Finally, we conducted a series of incremental ablation studies to examine the impact of dataset curation and pre-training configurations on representation performance in Tab. 6. All models are trained for 114.6K iterations. The results are summarized. The notation '+" indicates that each row introduces only one additional component relative to the previous configuration.

Table 6. **Downstream performance across various dataset and pretraining configurations**. All models were trained on a TRACKVERSE subset of size 392k for 300 epochs and evaluated on a range of downstream tasks.

	Downstream Task → / Pretrain Config ↓	LVIS-IN	TrackVerse		SSv2	MIT-States			
		NN	NN	FSL	Ft	Obj NN	Attr NN	Pair NN	Pair Ft HM
	392K-Random								
1	MoCo-v3 w/ Static frames	30.5	26.8	39.7	56.0	48.8	33.8	9.0	39.0
	392K-CB1000								
2	MoCo-v3 w/ Static frames	34.6	33.8	46.8	55.1	50.9	37.8	9.7	40.3
3	+ Temporal jitter	35.3	34.4	48.8	54.8	51.4	37.8	10.1	39.6
4	+ Time gap ($\delta = 2$)	36.7	36.9	49.7	55.7	52.4	38.6	10.4	39.7
5	+ Adaptive Gaussian blur	37.3	37.8	51.9	57.3	53.3	38.8	10.6	41.1
6	+ Aug. aware prediction	**38.4**	**39.9**	**53.1**	**57.6**	**54.7**	**39.3**	**10.9**	**41.1**

Pretraining Data (Method)	LVIS-IN NN	TrackVerse NN	TrackVerse FSL	SSv2 Ft	MIT-States Pair Ft.	Davis-2017 $\mathcal{J}\&\mathcal{F}_m$
Static Frames (MAE)	22.1	20.0	28.3	46.5	35.5	58.4
Tracks (MAE w/ Temporal Context)	**24.3**	**22.3**	**31.1**	**54.2**	**37.5**	**62.8**

Table 7. MAE Pretraining on TRACKVERSE: Static Frames vs. Temporally-Contextualized Tracks

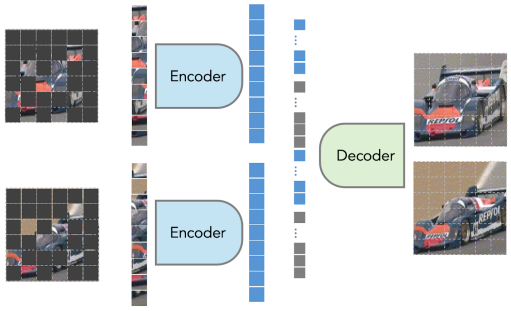


Figure 9. MAE w/ Temporal Context

D.2. Additional SSL methods

While our experiments focus on MoCo due to its reliability, we believe the findings are indicative of other contrastive methods as well. We also conducted additional experiments with Masked Image Modeling (MIM) to demonstrate TrackVerse's effectiveness with other SSL methods (beyond contrastive). To incorporate temporal augmentation, we modified the standard MAE (Fig. 9) by sampling two frames 2s apart, applying the same random mask, encoding them separately, and jointly reconstructing both. Trained on the 0.4M-CB1000 subset for 300 epochs, the results in Tab. 7 show that TRACKVERSE also benefits MIM-based approaches, supporting its broader applicability in different SSL methods.

E. Datasheet

In this section, we provide a detailed description of the dataset following the "datasheet for datasets" [4].

E.1. Motivation

For what purpose was the dataset created?

The TRACKVERSE dataset was created for self-supervised representation learning of common objects within their natural environments. This focus is designed to advance the understanding and development of algorithms capable of learning from dynamic and varied real-world conditions without the need for labeled data. The dataset supports the exploration of how objects change over time and interact with their surroundings, providing rich data that helps in developing more sophisticated and generalizable video-based SSL techniques that go beyond what is possible with static image data. The validation set is designed to evaluate the representations learned on TRACKVERSE in-domain.

E.2. Composition

What do the instances that comprise the dataset represent (*e.g.*, documents, photos, people, countries)?

The instances in the TRACKVERSE dataset represent object tracks extracted from YouTube videos. These tracks are centered around individual objects, with durations ranging from 3 to 30 seconds. Each object track is a sequence capturing a specific object's motion and changes over time, providing a focused view that supports detailed analysis and learning about the object's dynamics within its natural environment. Examples from TRACKVERSE can be seen in Fig. 2.

How many instances are there in total (of each type, if appropriate)?

The TRACKVERSE dataset comprises a total of 10.78 million object tracks in its full, uncurated form. The curated 1121K-CB2500 subset contains 1,121,000 object tracks. The validation set contains 4188 object tracks, with 698 object classes (6 tracks per class).

Does the dataset contain all possible instances or is it a sample (not necessarily random) of instances from a larger set?

The TRACKVERSE dataset contains all possible tracks that were found using a combination of the DETIC detector and the ByteTrack tracker, deployed using the LVIS object vocabulary. The proposed 1121K-CB2500 subset contains a subset that has been class-balanced for improved diversity within a fixed-size dataset.

What data does each instance consist of?

The TRACKVERSE dataset set is distributed as a GZIP compressed JSONL file with one instance per line. Each

instance is stored in JSON format, containing the following fields:

- `track_id`: unique track identifier
- `video_size`: [height, width] of the video from which this track was extracted
- `track_ts`: [start_time, end_time] timestamps (seconds) in the original video for the first and last frame in the track
- `top10_lbl`: Class ids of the top-10 predicted classes for the track, based on label score
- `top10_desc`: names of the top-10 predicted classes for the track, based on logits
- `top10_cls`: [[top-10 logits mean], [top-10 logits std]] A list of the mean values of the classification logits for the top 10 classes, and a list of the standard deviations for these logits
- `top10_wcls`: [[top-10 weighted logits mean], [top-10 weighted logits std]] A list of the mean scores for each of the top 10 weighted scores (weighted logits by the object track score), and a list of the standard deviations of these scores.
- `frame_ts`: timestamps (seconds) in the original video for each frame in the track
- `frame_bboxes`: list of bounding box coordinates [top_left_x, top_left_y, bottom_right_x, bottom_right_y] of the object for each frame in the track.
- `yid`: YouFube ID for the video from which this track was extracted
- `mp4_filename`: Filename of the track produced by running the track extraction pipeline.

Is there a label or target associated with each instance?

While the dataset set is proposed for the purpose of unsupervised representation learning, every instance in the TRACKVERSE dataset has been automatically categorized by the DETIC object detector. We further manually verified the labels of instances in a smaller validation set to ensure their correctness and reliability.

Is any information missing from individual instances?

No.

Are relationships between individual instances made explicit (*e.g.*, users' movie ratings, social network links)?

Yes, the spatiotemporal relationships between individual tracks in the TRACKVERSE dataset (*e.g.*, which tracks co-occurred in time) can be inferred from the temporal and spatial information provided with the dataset metadata. For each object track, we include timestamps and bounding boxes that detail the specific spatial location and duration of the object within the original video. This information establishes a clear contextual link between each track and its origin, allowing users to know not only the object's characteristics but also its interactions with other objects within the natural video environment.

Are there recommended data splits (*e.g.*, training, development/validation, testing)?

Yes, the full TRACKVERSE dataset can be curated into subsets of varying scales by capping the high-frequency classes at 100, 300, 500, 1000, and 2500 tracks per class, resulting in four curated subsets containing 82K, 184K, 259K, 392K and 1.12M tracks, respectively. For pretraining purposes, we recommend using the largest subset, the 1121K-CB2500, which offers a comprehensive range of data while maintaining class balance. Additionally, we provide a validation set that includes 4188 object tracks across 698 object classes, with each class represented by 6 tracks. The labels for these validation tracks were manually verified to ensure accuracy. We ensure that the validation set is disjoint from the training set to prevent data leakage and to provide a reliable benchmark for evaluating model performance.

Are there any errors, sources of noise, or redundancies in the dataset?

Yes, there are potential errors in the TRACKVERSE dataset stemming from the labels predicted by the DETIC open-vocabulary object detector, which are not 100% accurate. Further analysis of the data collection pipeline and failure modes are given in this supplement. Despite this, the primary purpose of TRACKVERSE is for video-based self-supervised learning, where labels are not utilized during the training process. The annotations primarily serve to enhance the dataset's diversity, ensuring it is more class-balanced and object-centric.

Is the dataset self-contained, or does it link to or otherwise rely on external resources (*e.g.*, websites, tweets, other datasets)?

The TRACKVERSE dataset is designed to be self-contained in terms of its primary data structure, but it relies on external resources for the video content itself. We plan to release only the URLs of the videos along with JSONL files containing metadata, instead of the videos directly. This approach is taken to respect user privacy, allowing the original uploaders the flexibility to delete or modify their videos. Users of the dataset should be aware that the availability and constancy of the video content linked via URLs cannot be guaranteed over time due to potential changes made by the video owners.

Does the dataset contain data that might be considered confidential (*e.g.*, data that is protected by legal privilege or by doctor-patient confidentiality, data that includes the content of individuals' non-public communications)?

No.

Does the dataset contain data that, if viewed directly,

might be offensive, insulting, threatening, or might otherwise cause anxiety?

No.

Does the dataset identify any sub-populations (*e.g.*, by age, or gender)?

Not explicitly.

Is it possible to identify individuals (that is, one or more natural persons), either directly or indirectly (that is, in combination with other data) from the dataset?

Yes, it is theoretically possible to identify individuals from the dataset as some videos may contain identifiable information like faces or license plates. To mitigate this risk, we have implemented measures to anonymize such information by using the Deface tool to blur all faces and license plates visible in the videos. Additionally, it is important to note that all videos included in the dataset are publicly available and were obtained in accordance with the Terms of Service agreed upon by users when uploading content to YouTube. These steps are part of our commitment to ensuring privacy and adhering to ethical standards in the use of publicly sourced video data.

Does the dataset contain data that might be considered sensitive in any way (*e.g.*, data that reveals race or ethnic origins, sexual orientations, religious beliefs, political opinions or union memberships, or locations; financial or health data; biometric or genetic data; forms of government identification, such as social security numbers; criminal history)?

The dataset primarily consists of video tracks from publicly available YouTube videos, which may inadvertently contain data that could be considered sensitive. This includes potential visibility of individuals' racial or ethnic origins, locations, or other personal characteristics due to the nature of the video content. However, no explicit collection of sensitive data such as sexual orientation, religious beliefs, political opinions, financial or health data, biometric or genetic data, government identification numbers, or criminal history was intended or conducted in the creation of the dataset. We employ measures such as blurring faces and license plates to minimize the risk of revealing identifiable information. Nonetheless, given the public source of the videos, residual risks of exposure to sensitive data may persist despite these precautions.

E.3. Collection Process

How was the data associated with each instance acquired? Was the data directly observable (*e.g.*, raw text, movie ratings), reported by subjects (*e.g.*, survey responses), or indirectly inferred/derived from other data (*e.g.*, part-of-speech tags, model-based guesses for age or language)?

We download the video data from YouTube. The videos are public and directly observable on YouTube.

What mechanisms or procedures were used to collect the data (*e.g.*, hardware apparatuses or sensors, manual human curation, software programs, software APIs)?

We provide a detailed dataset acquisition and tracking pipeline in the main paper. To summarize, we use the YouTube API and the youtube-dl library to download the original videos, PySceneDetect [2] to split the video into scene clips, a cartoon classifier and an aesthetics predictor to remove cartoons and low aesthetics videos, and DETIC [10] and ByteTrack [8] to detect and track objects in the video. We also cap the high-frequency classes to create more class-balanced subsets and manually verify the labels for the validation set.

If the dataset is a sample from a larger set, what was the sampling strategy (*e.g.*, deterministic, probabilistic with specific sampling probabilities)?

Each subset is a curated sample from the full TrackVerse dataset. The curation involves capping the high-frequency classes at predetermined thresholds—100, 300, 500, and 1000 tracks per class, resulting in four curated subsets containing 82K, 184K, 259K, and 392K tracks, respectively, to address and mitigate the issues of class imbalance often found in natural video dataset. Within each class, tracks are deterministically sampled based on the highest logit scores.

Who was involved in the data collection process (*e.g.*, students, crowdworkers, contractors) and how were they compensated (*e.g.*, how much were crowd-workers paid)?

The authors of this paper were involved in the data collection process. No crowdworkers or contractors were involved.

Over what timeframe was the data collected?

The data was collected over a period from January 2024 to May 2024. However, the YouTube videos included in the dataset may date back much earlier than this collection period.

Did you collect the data from the individuals in question directly, or obtain it via third parties or other sources (*e.g.*, websites)?

We collect the data from YouTube, which is a public platform.

Has an analysis of the potential impact of the dataset and its use on data subjects (*e.g.*, a data protection impact analysis) been conducted?

No formal analysis of the potential impact of the dataset on data subjects, such as a data protection impact analysis, has been conducted. Given that the dataset primarily comprises publicly available YouTube videos, the primary focus has been on adhering to general data privacy and usage guidelines, such as anonymizing identifiable informa-

tion where possible.

Is the software used to preprocess/clean/label the instances available?

E.4. Uses

Has the dataset been used for any tasks already?

No. The dataset is newly introduced in this paper.

What (other) tasks could the dataset be used for?

The evaluation set with verified labels can be used to assess video object recognition.

Is there anything about the composition of the dataset or the way it was collected and preprocessed/cleaned/labeled that might impact future uses?

The dataset was collected for studying video-based self-supervised learning of object representations as demonstrated in the main paper. While spatial relationships between objects are preserved in the dataset, they have not been explored in this work. Future works may explore the spatial relationships between objects in the dataset.

Are there tasks for which the dataset should not be used?

No.

E.5. Distribution

Will the dataset be distributed to third parties outside of the entity (*e.g.*, company, institution, organization) on behalf of which the dataset was created?

Yes.

How will the dataset be distributed (*e.g.*, tarball on website, API, GitHub)?

We distribute all the extracted track data (class labels, bounding box, timesteps, links to the original YouTube videos etc) as JSONL files and the code used for creating the dataset through GitHub.

Will the dataset be distributed under a copyright or other intellectual property (IP) license, and/or un- der applicable terms of use (ToU)?

Yes, under the permissible MIT license for research-based use only.

Have any third parties imposed IP-based or other restrictions on the data associated with the instances?

No.

Do any export controls or other regulatory restrictions apply to the dataset or to individual instances?

No.

E.6. Maintenance

Is there an erratum?

No, but it's important to note two issues: (1) the provided categories are DETIC predictions (not 100% accurate), and (2) some YouTube videos may be removed or made private by the owner.

Will the dataset be updated (*e.g.*, to correct labeling errors, add new instances, or delete instances)?

Yes, the dataset will be actively maintained and updated as needed. Updates to correct labeling errors, add new instances, or delete instances will be announced on the main webpage.

If the dataset relates to people, are there applicable limits on the retention of the data associated with the instances (*e.g.*, were the individuals in question told that their data would be retained for a fixed period and then deleted)?

No. We try to blur out all faces and license plates appearing in the video using Deface [1].

Will older versions of the dataset continue to be supported/hosted/maintained?

Yes, older versions of the dataset will continue to be hosted and maintained.

If others want to extend/augment/build on/contribute to the dataset, is there a mechanism for them to do so?

Yes, we will provide detailed instructions. Each submission will be carefully reviewed through GitHub's Pull Request system before merging it with the main dataset.

References

[1] Martin Bucella. Deface: Video anonymization by face detection. https://github.com/ORB-HD/deface. 11

[2] Brandon Castellano. Pyscenedetect. https://github.com/Breakthrough/PySceneDetect. 1, 10

[3] Xinlei Chen*, Saining Xie*, and Kaiming He. An empirical study of training self-supervised vision transformers. *ICCV*, 2021. 7

[4] Timnit Gebru, Jamie Morgenstern, Briana Vecchione, Jennifer Wortman Vaughan, Hanna Wallach, Hal Daumé Iii, and Kate Crawford. Datasheets for datasets. *Communications of the ACM*, 64(12):86–92, 2021. 8

[5] Raghav Goyal, Samira Ebrahimi Kahou, Vincent Michalski, Joanna Materzynska, Susanne Westphal, Heuna Kim, Valentin Haenel, Ingo Fruend, Peter Yianilos, Moritz Mueller-Freitag, et al. The" something something" video database for learning and evaluating visual common sense. In *ICCV*, pages 5842–5850, 2017. 6

[6] Phillip Isola, Joseph J. Lim, and Edward H. Adelson. Discovering states and transformations in image collections. In *CVPR*, 2015. 6

[7] F. Perazzi, J. Pont-Tuset, B. McWilliams, L. Van Gool, M. Gross, and A. Sorkine-Hornung. A benchmark dataset and evaluation methodology for video object segmentation. In *CVPR*, 2016. 4

[8] Yifu Zhang, Peize Sun, Yi Jiang, Dongdong Yu, Fucheng Weng, Zehuan Yuan, Ping Luo, Wenyu Liu, and Xinggang Wang. Bytetrack: Multi-object tracking by associating every

detection box. In *European conference on computer vision*, pages 1–21. Springer, 2022. 10

[9] Zhaoheng Zheng, Haidong Zhu, and Ram Nevatia. Caila: Concept-aware intra-layer adapters for compositional zero-shot learning. pages 1721–1731, 2024. 5

[10] Xingyi Zhou, Rohit Girdhar, Armand Joulin, Philipp Krähenbühl, and Ishan Misra. Detecting twenty-thousand classes using image-level supervision. In *European Conference on Computer Vision*, pages 350–368. Springer, 2022. 10