Building Scalable Video Understanding Benchmarks through Sports

Aniket Agarwal[1 †] **Alex Zhang**[2 †] **Karthik Narasimhan**[2] **Igor Gilitschenski**[3]
Vishvak Murahari[2 *] **Yash Kant**[3 *]

[1]**IIT Roorkee** [2] **Princeton University** [3]**University of Toronto**

[†]**Denotes equal contribution** [*]**Denotes equal advising**

https://asap-benchmark.github.io/

Abstract

Existing benchmarks for evaluating long video understanding falls short on two critical aspects, either lacking in scale or quality of annotations. These limitations arise from the difficulty in collecting dense annotations for long videos, which often require manually labeling each frame. In this work, we introduce an automated Annotation and Video Stream Alignment Pipeline (abbreviated ASAP). We demonstrate the generality of ASAP by aligning unlabeled videos of four different sports with corresponding freely available dense web annotations (*i.e.* commentary). We then leverage ASAP's scalability to create LCric, a large-scale long video understanding benchmark, with over 1000 hours of densely annotated long Cricket videos (with an average sample length of ∼50 mins) collected at virtually zero annotation cost. We benchmark and analyze state-of-the-art video understanding models on LCric through a large set of compositional multi-choice and regression queries. We establish a human baseline that indicates significant room for new research to explore. Our human studies indicate that ASAP can align videos and annotations with high fidelity, precision, and speed. The dataset along with the code for ASAP and baselines will be publicly released.

1 Introduction

Humans learn and master skills (*e.g.* playing guitar) by associating and reasoning over episodic memories captured over days, months, and years of failed and successful attempts (Byrne, 2008). Thus, building systems capable of understanding and reasoning over very long streams of visual data is a long-standing and crucial problem in Computer Vision.

Long-horizon Video Understanding (LVU) is the problem of reasoning over a long stream of video data, such as understanding the plot of a movie or analyzing the performance of a player in a lengthy game. Progress toward LVU has been greatly limited by the lack of densely annotated data. Creating an LVU benchmark requires manually annotating videos frame-by-frame, which is incredibly tedious and hard to scale. This constraint has limited the length of existing densely-annotated video understanding benchmarks (Table 1) from a few seconds (Jang et al., 2017; Sigurdsson et al., 2016; Gupta et al., 2021; Xu et al., 2016) to a few minutes (Krishna et al., 2017; Wu and Krahenbuhl, 2021; Zhou et al., 2018; Gella et al., 2018; Bain et al., 2020).

A line of previous works (Huang et al., 2020; Tapaswi et al., 2016; Lei et al., 2018) in LVU have used readily available subtitles of TV shows or entire movies as dense annotations. While these videos are sufficiently long, manual annotations are still required to build non-trivial queries to evaluate LVU skills (Tapaswi et al., 2016; Lei et al., 2018), which greatly limits their scale. Another work (Wu and Krahenbuhl, 2021) addresses this problem by extracting supervision from easily accessible YouTube metadata of nearly ∼30K movie clips spanning $1-3$ minutes. However, the annotated clips are short, and the proposed prediction tasks rely on noisy (and obscure) attributes (*e.g.* YouTube views, like-to-dislike ratio, and so on).

Sports matches are a rich source of long videos (*e.g.* a one-day Cricket match lasts nearly 8 hours) and usually have a brief scorecard embedded in the screen (shown in Figure 1) that tracks the state of the match. Most sports matches also have dense annotations from experts available online (sports commentary describing major events in the game *e.g.* (ESPN, 2022a,b)). However, the annotations or the videos are not helpful individually unless they are aligned with each other.

Therefore, we introduce ASAP, an automated annotation and video stream alignment pipeline, to auto-

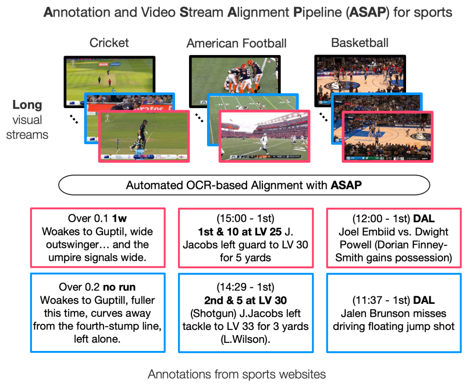

Annotation and Video Stream Alignment Pipeline (ASAP) for sports
Cricket
American Football
Basketball
Long visual streams
Automated OCR-based Alignment with ASAP
Over 0.1 1w Woakes to Guptil, wide outswinger… and the umpire signals wide.
(15:00 - 1st) 1st & 10 at LV 25 J. Jacobs left guard to LV 30 for 5 yards
(12:00 - 1st) DAL Joel Embiid vs. Dwight Powell (Dorian Finney-Smith gains possession)
Over 0.2 no run Woakes to Guptill, fuller this time, curves away from the fourth-stump line, left alone.
(14:29 - 1st) 2nd & 5 at LV 30 (Shotgun) J.Jacobs left tackle to LV 33 for 3 yards (L.Wilson).
(11:37 - 1st) DAL Jalen Brunson misses driving floating jump shot
Annotations from sports websites

Figure 1: **Overview.** We propose the **ASAP** pipeline for sports to align unlabelled sports videos (above) with structured annotations (below) publicly available on the web using an OCR based module. These newly aligned annotations can help compose structured queries that test long-horizon video understanding skills.

matically generate video datasets with frame-aligned dense annotations (i.e natural language commentary/major events) for arbitrarily long sports matches that have commentary freely available on the web (ESPN, 2022a,b)). Web annotations have an associated time marker like a game clock, which we call the *match state*, that ASAP extracts from sports match videos using an OCR detector (Google, 2022) for frame-by-frame annotation. To demonstrate the generality of ASAP, we align unlabeled videos of four distinct sports (Cricket, Football/Soccer, Basketball, and American Football) with their corresponding unaligned web annotations, with an average of 95% of the annotations being aligned within ±1 second of their occurrence in the video.

We then leverage ASAP's scalability to create LCric, a large-scale LVU benchmark with 1008 hours of densely annotated Cricket videos at virtually zero annotation cost, by auto-labeling 131 cricket matches of average length 7.5 hours, containing nearly 475 timestamp recordings (balls per match) on average. To our knowledge, LCric is the first automatically labeled sports video dataset that contains play-by-play annotations that span entire matches. To comprehensively evaluate LVU on LCric, we automatically curate multiple-choice (binary and N-way) and regression queries through simple composition with boolean operations, which require varying lengths of context to answer. These queries are complex and require context aggregation ranging anywhere from 5 minutes to an hour of continuous playtime (video). In the past, such compositional query building has been leveraged in popular vision and language datasets (*e.g.* CLEVR (Johnson et al., 2017), GQA (Hudson and Manning, 2019)).

We benchmark two recent state-of-the-art LVU models, TQN (Zhang et al., 2021) and MemViT (Wu et al., 2022), on LCric, and find that their performance is significantly worse than the human baseline (∼38% drop on query reasoning accuracy when evaluated on long clips containing ∼50 minutes of playtime), demonstrating significant room for new research to explore. In summary, we make the following contributions:

- We propose ASAP, an automated and scalable video labeling pipeline for aligning videos of sports matches of four different sports (Cricket, Football, Basketball, and American Football) with dense web annotations.
- Using ASAP, we create LCric, a large-scale LVU benchmark with 1008 hours of densely annotated Cricket videos with virtually zero annotation cost.
- We benchmark the performance of two recent video understanding models on our dataset, provide ablations, and establish a human baseline on LCric.

2 Related Work

Existing benchmarks for LVU. The paper Wu and Krahenbuhl (2021) introduces the large-scale LVU benchmark built on movie clips and metadata publicly available on YouTube. However, the videos only range from 1-3 minutes, and the annotations are limited due to their dependence on YouTube metadata. Other benchmarks for LVU include Oh et al., which collected 29 hours of surveillance footage and bounding box annotations of major events but only have clips of length up to 3 minutes. Similarly, Corona et al. (2021) collected 144 hours of surveillance footage by hiring actors to enact predefined scripts but only has clips of length up to 5 minutes. The Li et al. (2020) benchmark collected 430 videos, each 15 minutes long, and dense bounding box information for 80 different atomic actions. Although their videos are relatively long, the annotated videos are only up to 15 minutes long, which is shorter than our annotated videos which are up to 45 minutes long, and are generated with no additional cost. Additionally, Cheng-Yang Fu and Berg (2017) collected the LoL dataset comprising 230 clips

Dataset	Average clip length	# Annotations	# Hours	Auto labelled
VidSitu Gupta et al. (2021)	10 secs	145K	81	✗
VideoStory Gella et al. (2018)	18 secs	123K	396	✓
MSR-VTT Xu et al. (2016)	20 secs	**200K**	41	✗
Charades Sigurdsson et al. (2016)	30 secs	28K	82	✗
TGIF Jang et al. (2017)	30 secs	126K	86	✓
TVQA Lei et al. (2018, 2020)	75 secs	152K	460	✗
VTW Zeng et al. (2016)	90 secs	45K	213	✓
MovieClips Bain et al. (2020)	120 secs	30K	**1270**	✓
LVU Wu and Krahenbuhl (2021)	120 secs	11K	**1270**	✓
YouCook II Zhou et al. (2018)	316 secs	15K	176	✗
ActNet Captions Krishna et al. (2017)	180 secs	100K	849	✗
LCric (ours)	**2778 secs**	62K	1008	✓

Table 1: **Dataset Comparison.** Different annotated datasets for benchmarking video description and video understanding methods. LCric has an average clip length of ∼2800 seconds, which is almost ten times longer than any previous work.

from the League of Legends video game, with each clip ranging from 30 to 50 minutes. However, they collected video highlight annotations based on very noisy and unreliable audience chat statistics. Video games also tend to have easy visual cues before major highlights that incentivize models to learn spurious correlations. In contrast, our tasks, by construction, force models to reason over a long horizon of events in a match.

Collecting dense annotations for videos. Annotating video datasets is extremely expensive. Prior works (Xu et al., 2016; Gupta et al., 2021; Sigurdsson et al., 2016) have collected expensive annotations through Amazon Mechanical Turks (AMT) to label their clips with an associated text description, which greatly limits their scale (Table 1). Another line of work bootstraps from pre-existing annotations to generate new annotations. Other prior works bootstrap from pre-existing annotations to generate new annotations. For example, Bain et al. (2020) use existing captions on YouTube and IMDb metadata to label 30000 movie clips, but assume these labels span the entirety of their clips. Similarly, Zeng et al. (2016) take user-generated titles as labels for 18100 user-generated clips, but again assume that these labels span the entirety of their clips. While Gella et al. (2018) temporally align sentences from paragraph captions to social media videos to form annotated clips. In contrast, our dataset is densely annotated by temporally aligning publicly available sports annotations, which offer more structure than text descriptions and are therefore hierarchically composable, enabling the creation of queries that require large but dense context. Although (Liang et al., 2010b,a; Xu et al., 2006) also align sports videos to online commentary information, they use heuristic methods that are not as accurate and do not scale well for generating longer and more video matches.

Video datasets based on sports. Recent interest in using computer vision to drive sports analytics (Tuyls et al., 2021) suggests the importance of a dense annotation pipeline for sports videos. Current methods for producing sports datasets involve some form of manual annotations like in Safdarnejad et al. (2015), where they manually label 4100 sports clips based on the given action. Several works (Voeikov et al., 2020; Andriluka et al., 2018; Kazemi and Sullivan, 2012) have used automatically generated, densely labeled pose annotations for sports videos but are not easily scalable because they run computationally expensive, frame-level models to generate their annotations. Larger datasets such as Soomro et al. (2012); Karpathy et al. (2014) exist but primarily focus on action recognition over a single clip, rather than a full sports video. Our dataset focuses on producing play-by-play annotations spanning entire sports match. Our general annotation pipeline can be easily extended to creation of video datasets for other sports.

Video understanding models. Processing long videos is challenging, as it requires aggregating context over long horizons with limited computational and memory budgets. In SlowFast networks Feichtenhofer et al. (2019), they use a dual pathway operating at a low and high frame rate to enable the aggregation of context over longer horizons while capturing low-level visual attributes. Meanwhile, Feichtenhofer

(2020) introduce a simple technique for progressive architecture expansion (along axes such as temporal, depth, width, etc.), inspired by feature selection in machine learning to achieve efficient models. Taking advantage of the implicit nature of transformers to handle long-range data Bertasius et al. (2021) proposes to adapt the standard transformer architecture for videos by enabling spatiotemporal feature learning directly from a sequence of frame-level patches. In MeMViT (Wu et al., 2022), they introduce a memory-augmented multi-scale vision Transformer, greatly improves temporal support with minimal memory overhead, and achieves state-of-the-art performance on a variety of video understanding benchmarks. While the trend shows the model's capacity to handle longer and longer video clips more efficiently, the absence of a truly long-horizon dataset inhibits a fair comparison between these baselines and also inhibits the model's transferability to real-world video understanding tasks.

Automated annotation pipelines. Automating annotation pipelines, even partially, is critical to developing large-scale datasets. For example, the SBU dataset (NeurIPS, 2011) for image-text retrieval pruned and paired Flickr queries with a set of images, the Conceptual Captions dataset (Sharma et al., 2018) for image captioning leveraged the "Alt-text" HTML attribute in web images, and the RedCaps dataset (Desai et al., 2021) harvested 12 million image-text pairs from curated subreddits. Meanwhile, Pont-Tuset et al. (2020) partially automate their annotation pipeline and collect multi-modal image annotations by asking annotators to describe an image through audio while simultaneously hovering their mouse over the region they are describing. We hope that our fully automated pipeline, ASAP, will help create long and densely annotated video datasets at an unprecedented scale.

3 ASAP: Annotation and Video Stream Alignment Pipeline

Sports matches provide an abundant source of long videos, along with a rich source of corresponding play-by-play annotations (i.e. expert commentary of major events in the match) easily accessible on the web (ESPN, 2022a,b). These play-by-play annotations are, however, not useful standalone as they are not aligned with the video of the match. More formally, given a video with a set of frames $\mathcal{F}$, a set of events $\mathcal{A}$, and an associated set of *match state*'s $\mathcal{T}$, web annotations can be described as a known bijection $w : \mathcal{T} \to \mathcal{A}$. To align these annotations frame-by-frame means learning the mapping $a : \mathcal{F} \to \mathcal{A}$, which is a composition $a = (w \circ o)$, where $o : \mathcal{F} \to \mathcal{T}$ is the unknown frame-to-*match state* alignment function.

Our framework, which we call ASAP, automatically learns the alignment function $o : \mathcal{F} \to \mathcal{T}$ by parsing *match state* on each video frame (Sec. 3.1) using an Optical Character Recognition (OCR) detector after pre-processing noisy and occluded frames, as seen in Figure 3. We then use the *match state* and the known web annotations $w : \mathcal{T} \to \mathcal{A}$ (Sec. 3.2) to assign an event annotation to each frame (Refer to Appendix A.1 for a list of events). This multi-stage approach, as well tricks to reduce OCR calls, allows us to efficiently learn the alignment mapping $a : \mathcal{F} \to \mathcal{A}$. We note that sports were a powerful use-case of ASAP because scorecards are an intuitive *match state* with existing web annotation $w : \mathcal{T} \to \mathcal{A}$. Thus, ASAP enables the creation of long video datasets with unprecedented scale at virtually zero annotation cost. We describe the different stages of ASAP in more detail below.

3.1 Stage 1: Match State Extraction

Extracting Match State. In a sports video containing N video frames $[f_1, ..., f_N]$, we intuit that scorecards tend to have a fixed position on each frame, so we search for a tight bounding box containing the *match state* by first sampling a few frames uniformly throughout the video and running *OCR* on them. Next, we determine the bounding box where text changes gradually across frames (i.e the bounding box containing the scorecard) and crop every frame with this bounding box to reduce the complexity future *OCR* calls. Using these bounding boxes, we apply *OCR* over every frame to learn the function $o : \mathcal{F} \to \mathcal{T}$ that maps a frame $f_t \in [f_1, ..., f_N]$ to a *match state*. The choice of *match state* varies based on the available annotations: in Cricket, the *match state* is the *ball* that is currently being delivered (e.g. "30.5"), while in Football, the *match state* is the corresponding half and game clock (e.g. first half, $38 : 23$). We show example cropped scorecards in Figure 2.

Non-triviality of applying *OCR*. Under ideal circumstances, simply applying *OCR* over cropped video frames and scraping web annotations would make ASAP a complete pipeline. However, we find that in practice, locating the scorecard across frames, as well as extracting the correct *match state* is non-trivial

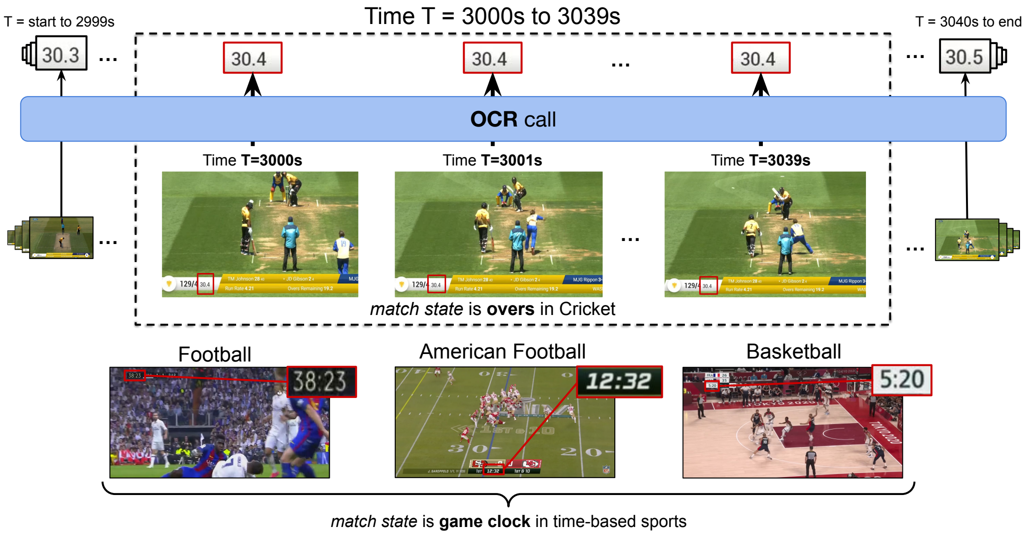
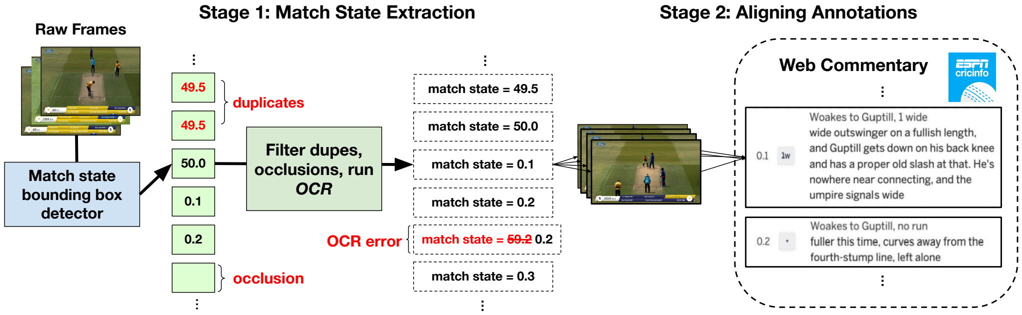


Figure 2: **Match States used by ASAP for different sports.** ASAP finds and stores the sequence of frames corresponding to each possible *match state*, which is necessary to temporally align web annotation data to its exact occurrence. For example, in Cricket, ASAP uses the overs number as the *match state*. For other sports like American Football, Football, and Basketball, ASAP uses the onscreen timer as *match state*. More generally, any temporal marker can be used as *match state*.



Woakes to Guptill, 1 wide
wide outswinger on a fullish length, and Guptill gets down on his back knee and has a proper old slash at that. He's nowhere near connecting, and the umpire signals wide

Woakes to Guptill, no run
fuller this time, curves away from the fourth-stump line, left alone

Figure 3: **Match State Extraction and Alignment.** To extract the match state we begin by detecting scorecard bounding box across all video frames, and cropping its content out. We then process all frames, filtering duplicates and outliers using an L1 distance metric, and finally run batched OCR calls to assign a *match state* to each frame for alignment with web commentary.

due to the dynamic and noisy elements that occlude, vary, or otherwise affect the long videos. Firstly, many sports matches contain advertisements, rendering glitches, and other occlusions that make the *match state* unreadable. To address these issues, ASAP extracts a *reference scorecard* before processing the whole video, which is a template image containing an un-occluded crop of the scorecard that the user can verify. We compare this reference image against all cropped scorecards in the video using a L1 distance metric, and throw out frames based on a heuristic thresholding value. Furthermore, *OCR* often makes mistakes due to noise or color variations in certain frames, so ASAP uses the temporally ordered nature of the *match state* to assign noticably incorrect mappings to the aligned *match state* of the previous frame. Finally, running *OCR* calls for every frame of hours of video footage running at 30 FPS can be prohibitively expensive. To reduce the costs many-fold, we stitch multiple scorecards into a single image to annotate hundreds of frames with a single `OCR` call. We also skip redudant frames that contain the same *match state* as an older frame by detecting changes in the scorecard between consecutive frames using an L1 distance metric similar to the reference image trick.

3.2 Stage 2: Aligning dense annotations with videos

Dense play-by-play annotations of the form $w : \mathcal{T} \to \mathcal{A}$ (i.e expert commentary, major events, etc.) are often easily available on the web (ESPN, 2022a,b). Since these play-by-play annotations are indexed by the *match state*, we map annotations based on their *match state* to their exact timestamps found in the first stage of ASAP (the function $o : \mathcal{F} \to \mathcal{T}$), which is precisely the function composition $a = (w \circ o)$ that describes a perfect mapping of each frame to an associated annotation.

In addition to aligning the annotations with the video, ASAP also processes the sequence of play-by-play annotations into a sequence of discrete events, which we refer to as an *event chain*. While some sports (Cricket and Football) already contain discrete events (e.g. 'foul', 'wicket', 'boundary' etc.) in their annotations, for other sports (American Football and Basketball), we use string-matching to parse the commentary and assign each play to a fixed event that we define (e.g. 'incomplete pass'). These extracted event chains can then be used as ground truth for evaluating LVU models. Models can be queried on different segments of the event chain of varying lengths – to test both short and long-horizon reasoning. We discuss the use of event chains for evaluation in Section 4.2. Finally, we demonstrate ASAP's generality by annotating other sports (Appendix A and Appendix B.5).

4 Generating the LCric dataset with ASAP

In this section, we describe how we leverage ASAP to build a long video understanding (LVU) dataset from Cricket videos. The ASAP pipeline takes in videos of Cricket matches along with play-by-play commentary annotations from the web[1] to aligh them together. Using the frame-aligned annotations produced by ASAP, we describe a scalable approach for generating structured and compositional queries in Section 4.2 to evaluate LVU. Our LCric dataset is a collection of Cricket videos with play-by-play annotations and a set of auto-generated queries. See Appendix B.1 for more details on the rules of Cricket.

4.1 LCric: Overview

Using ASAP's scalability, we create LCric, a large-scale LVU benchmark with 1008 hours of densely annotated Cricket videos at virtually zero annotation cost, by auto-labeling 131 cricket matches of average length ∼7.5 hours, containing nearly 475 timestamp recordings (balls per match) on average. ASAP automatically labels all the balls in a match with 1 of 12 events to generate a sequence of events (i.e *event chain*) for a cricket match. We then generate annotated video clips by segmenting the videos along with the aligned event chain into a contiguous sequence of 10-over (∼50 minutes) clips.

4.2 LCric: Testing LVU via compositional queries

In an LVU task, the evaluated model is given a very long video clip, from a sports match of nearly 50 mins in our case, and it is tasked to answer a question (query) about it, e.g. "how many times did a wide ball occur in this video?". An LVU system needs to possess two types of skills: a) the ability to reliably detect local (short-term) events – *e.g.*, classifying an atomic event in Cricket (say wide, wicket, or run), and b) the ability to aggregate information across these local events given a task (which we refer to as a query) – *e.g.*, counting the total number of runs scored by the batting team in an arbitrarily long video. We evaluate these LVU skills on LCric, by automatically generating binary, multiple-choice, and regression queries, and evaluating them on long video segments. Details on each type of query and how they were generated can be found in Appendix B.3.

4.3 LCric: Statistics and Dataset Splits

Statistics. LCric currently includes 1008 hours of cricket match videos across 131 unique matches (average length of 7.5 hours), along with 61957 ball-by-ball annotations. All the videos are preprocessed at a resolution of 360p and we provide links to the source videos of higher resolution.

Dataset splits To effectively test generalization, we split all the matches in LCric into train, validation, and test splits and ensure a 3:1:1 ratio of the number of hours in each split. Due to a limited compute availability, we present ablations on a subset of LCric and refer

[1] https://www.espncricinfo.com/

Model	Training	Binary Acc. ↑	Multiple Acc. ↑	Regression L1 Norm ↓
TQN	Mixed	57.68%	19.05%	17.21
MeMViT	Mixed	54.31%	16.71%	21.79
TQN	Homog.	**60.74%**	**20.19%**	**10.63**
MeMViT	Homog.	56.53 %	17.79%	11.95
Human		96.34 %	96.29%	0.215

Table 2: **Baseline performance on the full LCric split with 10 over clips.** TQN outperforms MeMViT in different training schemes across different query types. We find that training models under the Homogeneous (Homog.) training scheme improves performance, especially for the regression query.

to this as LCric-Mini, which has around 420 hours of labeled Cricket matches, and enables us to train ablations experiments in a shorter duration (2-3 days per experiment). We generate splits for LCric-Mini identically to LCric.

5 Experimental Setup and Results on LCric

Our preprocessing follows the process in (Zhang et al., 2021) by sampling videos at a lower frame rate to make training over long videos feasible. We process our longest clips (containing 10 overs of the match) at 0.1 FPS, and process clips of 2-8 overs at 0.5 FPS. We remove the scorecard from all frames to prevent annotation leakage and process frames at a resolution of 128 x 128. We compute following evaluation metrics: **1)** Classification accuracy for binary (*Binary Accuracy*) and multi-choice (*Multiple Accuracy*) queries *2)* Average L1 norm for regression queries (*Regression L1 Norm*).

5.1 Baselines and Training Scheme

Previous works (Fan et al., 2021; Feichtenhofer et al., 2019; Feichtenhofer, 2020) in LVU use pretrained CNNs (LeCun and Bengio, 1998) and Transformers (Vaswani et al., 2017) paired with explicit memory modules for modeling long contexts. However, none of these methods can scale to video clips longer than a few minutes. Since our query set requires reasoning over contexts ranging up to an hour, we choose two state-of-the-art video understanding models:

Memory-augmented Multiscale Vision Transformer (MeMViT) (Wu et al., 2022) applies a memory caching strategy by processing videos in an online fashion, allowing the model to efficiently store context to reason over long horizon. MeMViT builds upon ViT (Dosovitskiy et al., 2021) by using a novel pooling method and a dynamic patch resolution approach to reduce computational costs while processing long clips. We adapt MeMViT to handle our multi-query setting by leveraging the multi-query head used in TQN.

Model	Clip Segments	Binary Accuracy ↑	Multiple Accuracy ↑	Regression L1 Norm ↓
TQN	GT	**80.34%**	**36.32%**	**6.89**
TQN	Uniform	60.29%	15.18%	19.31
Human		96.34 %	96.29%	0.215

Table 3: **Both localization and detection of events are important.** Using ground truth (GT) clip segments (aligned by ASAP) for event prediction leads to significant performance gain (Rows 1 vs 2). Improving event detection within clips can further improve performance (Rows 2 vs 3).

For both of these baselines, we employ two different training schemes – *1) Homogeneous training* where we train different models for the three different types of queries (binary, multi-choice, and regression) and *2) Mixed training* where we train a single model for all three types of queries.

Human Baseline. To quantify the room for modeling improvements on LCric, we measure the accuracy of human annotators through AMT (Crowston, 2012). We provide annotators with video clips from LCric and ask them to predict the sequence of ball-by-ball events (Section 4). To compute human performance on our queries, we assume that given an event chain, humans can answer these queries by applying logical operators without mistakes. We detail our AMT setup in Appendix B.5.

5.2 Key Results

Performance of both TQN and MeMViT degrades rapidly for very long clips. To understand the impact of length of the videos on task performance, we train different baseline models for clips with over-lengths ranging from 2 overs (∼10 minutes) to 10 overs (∼50 minutes). Figure 4 shows that performance rapidly decreases with increasing clip length and approaches the random baseline for binary and multi-choice queries. This result, in addition to the strong human baseline, shows significant room for modeling improvements.

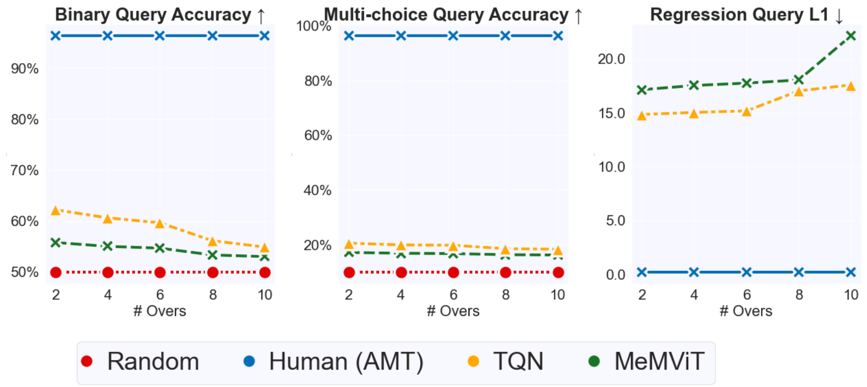


Figure 4: **Performance drops as video length (# of overs) increase.** We plot the accuracy achieved by our baseline models for binary, multiple-choice, and regression queries across clips varying from 2-10 overs (length). LVU models severely degrade in performance with longer time horizons and perform much worse than humans.

Models need to localize and detect events accurately to perform well on LCric. To understand the importance of localizing and detecting events for LVU models, we first train a TQN 'event classifier' model to predict 1 of 12 events (Section 4) in a video clip. The model is trained on ground truth annotations from LCric-Mini and has a fairly high test accuracy of 84.79%. We then divide the clips into a contiguous sequence of 'event segments' by either using ground truth segmentations from LCric-Mini (labeled 'Ground Truth' in Table 3) or by uniformly dividing the clips into 60 contiguous segments (labeled 'Uniform' in Table 3), as each 10-over clip contains 60 events. Finally, we leverage the learned 'event classifier' model to generate event chains by sequentially predicting events on the 'event segments' and evaluating different queries on these event chains. We report performance in Table 3 and make two observations – **1)** Access to ground truth event segments leads to $\sim 20\%$ improvement over uniformly generated segments on binary queries and therefore shows the importance of event localization. **2)** While access to ground truth event segments leads to better performance, as it aids the event classifier in making more accurate predictions, the performance is still $\sim 16\%$ worse than the human baseline on binary queries. Therefore, even with perfect event localization, models need performant event detection capabilities.

6 Conclusion

In this work, we introduce ASAP, a fully automated annotation and video stream alignment pipeline for sports matches. ASAP automatically aligns unlabeled videos of sports matches with corresponding dense annotations (*i.e.* commentary) freely available on the web. We demonstrate the generality of ASAP by aligning unlabeled matches of four very different sports with their corresponding annotations on the web. ASAP is highly accurate (as judged by human annotators), and is robust to varying visual attributes, number of events, and length of plays. We then demonstrate ASAP's potential to generate large-scale video datasets with *no additional annotation cost* by generating LCric, a large-scale long video understanding benchmark with over 1000 hours of densely annotated long Cricket videos (having an average sample length of ~50 minutes). We extensively benchmark state-of-the-art LVU models and establish a human baseline on LCric. Our strong human baseline, coupled with the poor performance of state-of-the-art models, validates LCric as an effective benchmark for the next generation of LVU models. We hope that future work extends, improves, and leverages ASAP to generate annotated video datasets at an unprecedented scale and cost efficiency. We also include a Datasheet adhering to Gebru et al. (2021) in Appendix C.2.

7 Reproducibility Report

ASAP Code: An anonymized version of our codebase for our ASAP pipeline described in the main paper, along with the related setup instructions can be found at the link here: https://github.com/asap-benchmark/asap-pipeline. The repository contains other details for running different parts of the pipeline on different sports.

LCric Dataset: We have provided a downloader for the LCric dataset, including both the related videos and annotations: https://github.com/asap-benchmark/lcric-downloader. Further implementation details and also details about LCric can be found in Appendix B.

Baseline Experiments: We also provide implementations used for running our baseline experiments here: https://github.com/asap-benchmark/lcric-baseline. The training schemes and experimental settings can be seen in Section 5.

References

Mykhaylo Andriluka, Umar Iqbal, Eldar Insafutdinov, Leonid Pishchulin, Anton Milan, Juergen Gall, and Bernt Schiele. Posetrack: A benchmark for human pose estimation and tracking. In *CVPR*, 2018.

Max Bain, Arsha Nagrani, Andrew Brown, and Andrew Zisserman. Condensed movies: Story based retrieval with contextual embeddings. In *ACCV*, 2020.

Gedas Bertasius, Heng Wang, and Lorenzo Torresani. Is space-time attention all you need for video understanding? In *ICML*, 2021.

John H. Byrne. *Learning and Memory: A Comprehensive Reference*. Elsevier, 2008. URL https://www.sciencedirect.com/science/referenceworks/9780123705099.

Mohit Bansal Cheng-Yang Fu, Joon Lee and Alexander C. Berg. Video highlight prediction using audience chat reactions. In *EMNLP*, 2017.

Kellie Corona, Katie Osterdahl, Roderic Collins, and Anthony Hoogs. Meva: A large-scale multiview, multimodal video dataset for activity detection. In *WACV*, 2021.

Kevin Crowston. Amazon mechanical turk: A research tool for organizations and information systems scholars. In Anol Bhattacherjee and Brian Fitzgerald, editors, *Shaping the Future of ICT Research. Methods and Approaches*, 2012.

Karan Desai, Gaurav Kaul, Zubin Aysola, and Justin Johnson. RedCaps: Web-curated image-text data created by the people, for the people. In *NeurIPS-DB*, 2021.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *ICLR*, 2021.

ESPN. Espncricinfo, 2022a. www.espncricinfo.com/.

ESPN. Espn soccer commentary, 2022b. https://www.espn.in/football/commentary.

Haoqi Fan, Bo Xiong, Karttikeya Mangalam, Yanghao Li, Zhicheng Yan, Jitendra Malik, and Christoph Feichtenhofer. Multiscale vision transformers. In *ICCV*, 2021.

Christoph Feichtenhofer. X3d: Expanding architectures for efficient video recognition. In *CVPR*, 2020.

Christoph Feichtenhofer, Haoqi Fan, Jitendra Malik, and Kaiming He. Slowfast networks for video recognition. In *ICCV*, 2019.

Timnit Gebru, Jamie Morgenstern, Briana Vecchione, Jennifer Wortman Vaughan, Hanna Wallach, Hal Daumé III au2, and Kate Crawford. Datasheets for datasets. *arXiv preprint*, 2021.

Spandana Gella, Mike Lewis, and Marcus Rohrbach. A dataset for telling the stories of social media videos. In *EMNLP*, 2018.

Google. Google cloud optical character recognition, 2022. https://cloud.google.com/vision/docs/ocr.

Arka Gupta, Mark Yatskar, Ram Nevatia, and Aniruddha Kembhavi. Visual semantic role labeling for video understanding. In *CVPR*, 2021.

Qingqiu Huang, Yu Xiong, Anyi Rao, Jiaze Wang, and Dahua Lin. Movienet: A holistic dataset for movie understanding. In *ECCV*, 2020.

Drew A Hudson and Christopher D Manning. Gqa: A new dataset for real-world visual reasoning and compositional question answering. In *CVPR*, 2019.

Yunseok Jang, Yale Song, Youngjae Yu, Youngjin Kim, and Gunhee Kim. Tgif-qa: Toward spatio-temporal reasoning in visual question answering. In *CVPR*, 2017.

Justin Johnson, Bharath Hariharan, Laurens Van Der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *CVPR*, 2017.

Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. Large-scale video classification with convolutional neural networks. In *CVPR*, 2014.

Vahid Kazemi and Josephine Sullivan. Using richer models for articulated pose estimation of footballers. In *BMVC*, 2012.

Ranjay Krishna, Kenji Hata, Frederic Ren, Li Fei-Fei, and Juan Carlos Niebles. Dense-captioning events in videos. In *ICCV*, 2017.

Yann LeCun and Yoshua Bengio. *Convolutional Networks for Images, Speech, and Time Series*. 1998.

Jie Lei, Licheng Yu, Mohit Bansal, and Tamara L Berg. Tvqa: Localized, compositional video question answering. In *EMNLP*, 2018.

Jie Lei, Licheng Yu, Tamara Berg, and Mohit Bansal. TVQA+: Spatio-temporal grounding for video question answering. In *ACL*, 2020.

Ang Li, Meghana Thotakuri, David A Ross, João Carreira, Alexander Vostrikov, and Andrew Zisserman. The ava-kinetics localized human actions video dataset. *arXiv preprint*, 2020.

Chao Liang, Yu Jiang, Jian Cheng, Changsheng Xu, Xiaowei Luo, Jinqiao Wang, Yu Fu, Hanqing Lu, and Jian Ma. Personalized sports video customization for mobile devices. In *MMM*, 2010a.

Chao Liang, Changsheng Xu, and Hanqing Lu. Personalized sports video customization using content and context analysis. In *IJDMB*, 2010b.

NeurIPS. Im2text: Describing images using 1 million captioned photographs. 2011.

Sangmin Oh, Anthony Hoogs, Amitha Perera, Naresh Cuntoor, Chia-Chih Chen, Jong Taek Lee, Saurajit Mukherjee, J. K. Aggarwal, Hyungtae Lee, Larry Davis, Eran Swears, Xioyang Wang, Qiang Ji, Kishore Reddy, Mubarak Shah, Carl Vondrick, Hamed Pirsiavash, Deva Ramanan, Jenny Yuen, Antonio Torralba, Bi Song, Anesco Fong, Amit Roy-Chowdhury, and Mita Desai. A large-scale benchmark dataset for event recognition in surveillance video. In *CVPR 2011*. doi: 10.1109/CVPR.2011.5995586.

Jordi Pont-Tuset, Jasper Uijlings, Soravit Changpinyo, Radu Soricut, and Vittorio Ferrari. Connecting vision and language with localized narratives. In *ECCV*, 2020.

Seyed Morteza Safdarnejad, Xiaoming Liu, Lalita Udpa, Brooks Andrus, John Wood, and Dean Craven. Sports videos in the wild (svw): A video dataset for sports analysis. In *FG*, 2015.

Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *ACL*, 2018.

Gunnar A Sigurdsson, Gül Varol, Xiaolong Wang, Ali Farhadi, Ivan Laptev, and Abhinav Gupta. Hollywood in homes: Crowdsourcing data collection for activity understanding. In *ECCV*, 2016.

Enxin Song, Wenhao Chai, Guanhong Wang, Yucheng Zhang, Haoyang Zhou, Feiyang Wu, Xun Guo, Tian Ye, Yan Lu, Jenq-Neng Hwang, et al. Moviechat: From dense token to sparse memory for long video understanding. *arXiv preprint arXiv:2307.16449*, 2023.

Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. UCF101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint*, 2012.

Makarand Tapaswi, Yukun Zhu, Rainer Stiefelhagen, Antonio Torralba, Raquel Urtasun, and Sanja Fidler. MovieQA: Understanding Stories in Movies through Question-Answering. In *CVPR*, 2016.

Karl Tuyls, Shayegan Omidshafiei, Paul Muller, Zhe Wang, Jerome Connor, Daniel Hennes, Ian Graham, William Spearman, Tim Waskett, Dafydd Steel, Pauline Luc, Adria Recasens, Alexandre Galashov, Gregory Thornton, Romuald Elie, Pablo Sprechmann, Pol Moreno, Kris Cao, Marta Garnelo, Praneet Dutta, Michal Valko, Nicolas Heess, Alex Bridgland, Julien Pérolat, Bart De Vylder, S. M. Ali Eslami, Mark Rowland, Andrew Jaegle, Remi Munos, Trevor Back, Razia Ahamed, Simon Bouton, Nathalie Beauguerlange, Jackson Broshear, Thore Graepel, and Demis Hassabis. Game plan: What ai can do for football, and what football can do for ai. *arXiv preprint*, 71, 2021. doi: 10.1613/jair.1.12505. URL http://dx.doi.org/10.1613/jair.1.12505.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017.

Roman Voeikov, Nikolay Falaleev, and Ruslan Baikulov. Ttnet: Real-time temporal and spatial video analysis of table tennis. In *CVPR-W*, 2020.

Chao-Yuan Wu and Philipp Krahenbuhl. Towards long-form video understanding. In *CVPR*, 2021.

Chao-Yuan Wu, Yanghao Li, Karttikeya Mangalam, Haoqi Fan, Bo Xiong, Jitendra Malik, and Christoph Feichtenhofer. Memvit: Memory-augmented multiscale vision transformer for efficient long-term video recognition. In *CVPR*, 2022.

C.S. Xu, J. Wang, K. Wan, Y. Li, and L. Duan. Live sports event detection based on broadcast video and web-casting text. In *Proceeding of ACM International Conference on Multimedia*, 2006.

Jun Xu, Tao Mei, Ting Yao, and Yong Rui. MSR-VTT: A Large Video Description Dataset for Bridging Video and Language. In *CVPR*, 2016. doi: 10.1109/CVPR.2016.571.

Kuo-Hao Zeng, Tseng-Hung Chen, Juan Carlos Niebles, and Min Sun. Title generation for user generated videos. In *ECCV*, volume 9906, 10 2016. ISBN 978-3-319-46474-9. doi: 10.1007/978-3-319-46475-6_38.

Chuhan Zhang, Ankush Gupta, and Andrew Zisserman. Temporal query networks for fine-grained video understanding. In *CVPR*, 2021.

Luowei Zhou, Chenliang Xu, and Jason J Corso. Towards automatic learning of procedures from web instructional videos. In *AAAI*, 2018.

Appendix A. ASAP

Accuracy and Speed of ASAP

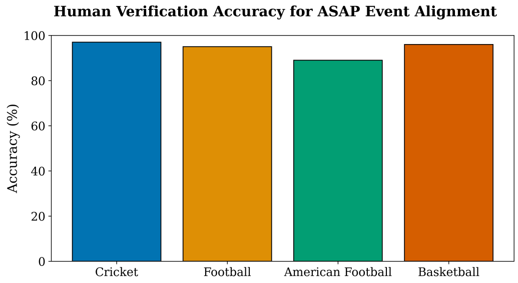


Figure 5: **Accuracy of ASAP.** Our human studies indicate ASAP alignment to be highly accurate, averagely 95% of the annotations are correctly aligned to the corresponding video events (±1 second) across 4 different sports.

To verify ASAP's ability to align dense annotations on the web with videos of sports matches, we conduct a study with human annotators on Amazon Mechanical Turk (AMT). We randomly sample clips from sports matches corresponding to 6 contiguous events in the event chain generated by ASAP. For all the generated clips, we ask human annotators to provide timestamps for all 6 events and then check whether the provided timestamps belong to the intervals generated for those events by ASAP. We plot the resulting accuracy of the timestamps in Figure 5 and find that ASAP is highly accurate, with an average accuracy of 95.3% across four very different sports, each differing in visual attributes, number of events, and length of plays. The drop in accuracy for American football annotations can be explained by the inconsistency in the timestamps provided by ESPN. For regular plays, the timestamp indicates when the play begins; however, for 'touchdowns', the timestamp indicates when the team scored and not when the play begins. Additionally, penalties may affect the game clock, which we use to align our annotations, sometimes leading to slight alignment issues, which we detail below. ASAP generates annotations with very high speed requiring just 10 minutes to align and process around 7 hours of video at 30FPS on a single machine.

Alignment Verification We verify the accuracy of annotations made by ASAP by providing human annotators with a clip containing a contiguous sequence of events, and asking them to provide the timestamps in the video for when each event occurred. Additionally, all scorecard information is masked in each provided clip.

Verification of different sports For cricket, we built an AMT interface and asked annotators to provide both the timestamps and events that occurred in a clip for over 1200 events to verify that both the ASAP alignment process and video quality were sufficient, which we discuss more in Appendix B. For verifying and demonstrating the generality of ASAP pipeline, we annotate three different sports, namely, American football, football, and basketball, and verify it using a similar interface. Due to the limited mturk budget, we used two of the in-house annotators for the verification of these three sport's annotations by providing the humans with clips from 6 hours of match footage for each sport and had them verify (by annotating) 240 events for each sport.

American Football Alignment Issues We note that the reason why the verification accuracy for American football in Figure 5 is lower than the other sports is because for most standard plays, the timestamps provided are for when the play started. However, when a team scores or is given a penalty, the timestamp provided for the next play is either the end of the play, or when it happened. We were only able to have ASAP account for the touchdown instances, but not the penalty instances, which is generally what was marked incorrect during our verification process.

A.1 Annotation Event Details

Events for Different Sports In this section, we describe the events that we considered for each sport.

- **Cricket:** Each legal delivery was considered a valid event, where features such as the number of runs and the occurrence of a wide/out ball were marked as well. See Appendix B for further details.
- **American Football:** Each play was considered a valid event, so we considered *punts, field goals, complete passes, incomplete passes, run-plays, sacks, penalties,* and *spikes* as distinct.
- **Football/Soccer:** There are no distinct, sequential plays in football, so we based our events off of online commentary. We mark *shots off target, shots on target, shots on woodwork, goals, fouls, substitutions, yellow cards, red cards, corner kicks, free kicks, offsides, handballs,* and

saved/blocked balls as distinct events to be annotated and aligned.

- **Basketball:** Like football/soccer, there are no distinct plays that happen, so we mark *fouls, jumper shots, layups, dunks, free throws*, and *regular shots* as distinct events that we annotate and align.

Granularity of Annotations Because the aligned annotations for different sports rely on the timestamps provided by the online commentary source, we observe that different sports are annotated with varying levels of granularity. Thus, when we verify the accuracy of an aligned annotation, we account for these differing levels of granularity with different margins for error. For example, in football, annotations are provided at a minute-level, so if the human annotator marks the event as occurring anywhere outside that range, we consider the annotation to be incorrect; however, for sports like basketball, where annotation timestamps are given by the second, we provide a margin of error of ± 1 second to the timestamp marked by the human. Similar to football, in cricket, an event lasts for 30-40 seconds, so if a human annotator is able to mark the event as occurring anywhere inside that range, we consider the annotation to be correct.

A.2 Raw Videos Source

All of the videos that we ran ASAP through were found across YouTube. For cricket we used 131 videos, and for all other three sports we annotated 3 videos each. The average video length of a cricket match is 7.5 hrs while for the other sports it is 1.5 hrs. We also provide the links to all the videos annotated with the supplementary document.

A.3 Examples of Frames filtered by ASAP

Some of the examples of frames being rejected by ASAP pipeline can be seen in Figure 6. As can be seen in most of these cases either the scorecard information is obstructed by some other text or some random data is present in its location.

A.4 Limitations of ASAP

ASAP focuses specifically on sports videos due to the nature of sports matches and the abundance of web annotations. Thus, it is difficult, though not impossible, to find online videos that satisfy the properties discussed in Section 3. Furthermore, the accuracy of ASAP's annotations depend on the accuracy and granularity of the web annotations and associated match states provided. In the case of sports videos, we have found that the time markers and annotations provided by sites like ESPN are extremely accurate, but this may not be the case for other domains. Finally, ASAP works the best for videos where the visible match state exhibits consistent visual properties throughout the video (e.g. a scorecard looks the same and does not move throughout the video). In instances where the match state is not in a fixed position throughout the video or has changing visual features, ASAP requires some manual effort.

Appendix B. LCric

B.1 Primer on Cricket [Video]

In this section we further extend our primer to Cricket provided in Section 4 by describing the Batting/Bowling phases, as well as the primary objective of the game. A brief overview video for the game explaining the game can be found in here.

Introduction to Cricket Cricket is a sport played by two teams of 11 players each that alternate between *batting* and *fielding* throughout the game. The batting team aims to score *runs* by hitting a ball bowled by the fielding team out of the playing field. Meanwhile, the fielding team aims to prevent the batting team from scoring runs and dismiss all players in the batting team by taking their *wickets*. Each exchange where the fielding team bowls a valid ball and the batting team attempts to hit the ball to score runs is called a *ball* (or *delivery*) and a sequence of 6 *balls* is called an *over*. Each *ball* is an *atomic event* and there are 12 distinct possible events. Those are: having the batting team score n runs ($n \in \{0, ..., 9\}$), a wicket is taken and the current batsman is dismissed, and a wide (invalid) ball is bowled giving the batting team an extra run and another ball. The game is played in an inning-format, where one team is batting, and the other team is fielding. We describe the two phases below.

Bowling Phase When a team is in the bowling phase, all 11 players stay on the field. One of the players is designated as the bowler, and their job is to deliver the ball to the batter (hitter) on the batting team. If the ball is struck by the batsman, the remaining players, called fielders, try to prevent the ball from reaching the boundary of the field and return the ball back to the pitch area. A single over consists of six deliveries

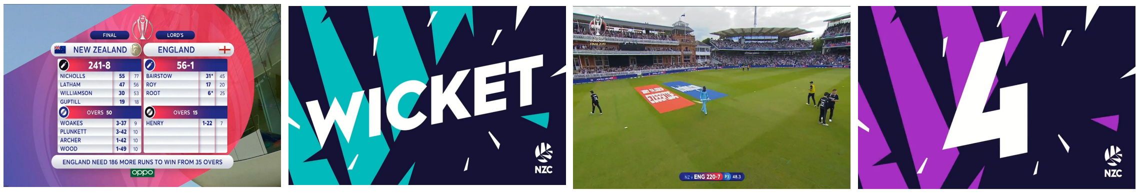


Figure 6: Some examples of frames rejected by ASAP. In all of these frames the scorecard information either get obstructed by screen overlays or shifted from their usual position.

bowled by the same player, and each team delivers a set number of overs depending on the tournament type in their bowling phase.

Batting Phase When is team is in the batting phase, only two players on the team stay on the field at a time. The batsman's job is to score runs and defend their wickets. A single run is scored when the batsman hits the ball and runs from one end of the pitch to another. Another way to score runs is to hit the ball to the boundary of the field, which is called the **'boundary'**, giving 4 or 6 runs to the batting team. In total, each batting team has 10 wickets.

Objective During an inning, the batting team wants to score as many runs as possible, while the bowling team wants to take as many wickets as possible to stop the batting team from scoring. In most single-day matches, the bowling team will bowl for 50 overs before the teams switch roles for the second half of the game. At this point, the goal of the new batting team is to outscore the previous team in runs before 50 overs or before losing all of their wickets.

B.2 Training and Implementation Details

We use consistent training schemes for both TQN Zhang et al. (2021) and MeMViT Wu et al. (2022) to provide a fair comparison between the two baselines. Both models were trained for 50 epochs on 4 V100 GPUs with a batch size of 4. We used a base learning rate of $LR = 0.01$ with the Adam optimizer and default hyperparameters.

Baseline Implementations For setting up TQN as a baseline, we used the official code provided by the authors with some minor modifications to the output heads for answering LCric queries. For MeMViT, since there is no official implementation released at the time of writing, we implemented our own version using the same implementation details as the main paper. Our implementation is built on top of the official implementation of MViT Fan et al. (2021), which is the base model used to create MeMViT.

Human Baseline Details To measure human performance, we conducted an AMT study on our LCric test set where we gave the annotators a set of possible events in Cricket (which is also what we give to the video understanding baselines), and asked them to both annotate the timestamps of these events and annotate the event that occurred (see supplementary). We assumed that humans possess near-perfect aggregation skills (i.e. if we ask a human to count the number of goals in a clip, and the human recognizes that a goal occurred in two different parts of the clip, the human can reasonably infer that a total of two goals occurred) and therefore we aggregated their answers on the clips to the queries generated for LCric.

B.3 LCric Queries

The following section describes the different types of queries automatically generated for the LCric dataset.

Min-Max occurrence query. To test a model's ability to detect and remember events, we construct queries such as "for a given video, did a wide ball (an event) occur between 3 and 5 times inclusive?". We generate these queries by sampling an atomic event from the set of all possible events, and then sampling two numbers, o_{min} and o_{max}, to denote the minimum and the maximum number of occurrences needed for this query to be *true*.

Binary queries by chaining occurrence queries. To increase query diversity and complexity, we sample n_{chain} different min-max occurrence queries and combine them using [and]/[or] operators. For example, for a given video spanning 10 overs (∼50 mins), "did a wide ball occur between 3 to 5 times [and] did a ball with 2 runs scored occur 1 to 3 times?". All *binary* queries in LCric are formed by chaining 1-5 different min-max occurrence queries.

Multiple-choice queries by counting occurrences. We expand upon the binary occurrence queries by generating multiple-choice occurrence queries, which ask models to directly predict the number of occurrences rather than predicting membership in a range. An example of such a query is – given a video, how many times did a wide ball occur after a ball with 4 runs? We note that these events are sequential, but not necessarily contiguous. As most non-trivial multiple-choice events in LCric occur between 0-9 times in a given clip, we use $\{0, ..., 9\}$ as our answer choices.

Filtering unbalanced queries. We can compose many LVU multiple-choice and binary queries using the above formulation, however, not all queries are balanced. Due to the rarity of certain events occurring in Cricket, some queries are far easier to guess correctly than others. For example, in a 45 minutes clip (spanning $\sim$ 10 overs), the query – "did a ball with 2 runs occur between 0 to 10 times" is true with a probability of 87%. We filter such queries based on the probability of their occurrence in training matches and ensure the average probability of occurrence of the selected queries to be between $0.45 - 0.55$ to avoid bias.

Regression query for counting runs. Lastly, we also experiment with a single regression query that asks the model to predict the number of runs scored as a regression output for a given video sequence.

Query Set Generation Algorithm We describe our query set generation process in Algorithm 1, where we use logical operators and a set of possible atomic events form form different combinations of queries.

Binary Query Statistics For our 10-over experiments, we formed a balanced set of 32 queries by taking queries from the set formed by Algorithm 1 and pruning them down so that given a random 10-over clip sampled uniformly from LCric, there would be a 0.5 ± 0.05 probability that the query would hold true on that clip. We list the set of all such queries and their corresponding probabilities in Table 4.

Multi-Choice Query Statistics We also generated a set of multi-choice queries for our 10-over experiments. These queries include a mix of common and less common event chains that generally occur between 0-9 (inclusive) times within any 10-over clip.

Algorithm 1: Query Set Generation

```
# The set of atomic events:
  [0,1,2,...,9,W,w]
Set of atomic events: $A_e$
# The number of queries for the query set
Size of the query set: $n_q$
query_set = []
for i in range($n_q$) do
    # Step A: getting raw operators and
      combinators choice
    num_joins ~ [1,5]
    # total length for operators set being
      sampled
    for determining the query length
    ops = random.choices([atleast(),
      atmost(), inrange()], num_joins)  #
      sampling list of operators
    combine_op = random.choices([and, or], 1)
      # sampling the combination operator
    # Step B: instantiating a query for
      query set for matching
    query = []
    for op in ops do
        # specify lower bound for
          atleast/inrange ops
        occ_min ~ [1,10]
        # specify upper bound bound for
          atmost/inrange ops
        occ_max ~ [occ_min,10]
        # sample atomic events in query
        atomic_event ~ [$A_e$]
        # Using the above variables for
          defining an occurrence pattern for
          atomic_event
        instanced_op = op(occ_min, occ_max,
          atomic_event)
          query.append(instanced_op)
    final_query = join_op(query)
    query_set.append(final_query)
```

The frequency of occurrence of these clips within our train set is provided in Figure 12.

B.4 LCric Statistics

We also plot additional statistics for LCric in Figure 7 and Figure 8. Figure 7 shows the distribution of occurrence of various clip lengths for 1-over, 5-over and 10-over. We clearly see the 10-over mark having more clips in 2000-3000 secs time range while the 1-over mark has more clips in 200-300 time range. Figure 8 also shows the number of clips for different over marks.

B.5 AMT Interface

We built an AMT interface for verifying ASAP's alignment of cricket annotations to videos, with the full instructions and interface provided in Figure 9.

Instruction Details Each annotator is given a set of instructions to read prior to beginning the main annotation task, called a HIT (Human Intelligence Task). For each task, the annotator is given a video clip from a sports match. The task is to classify each legal delivery/ball that occurred in the video, as well as the timestamp at which the annotator was able to gather enough information to answer this question. Additionally, we provide a set of examples for what each event looks like to the annotators, as well as a fully annotated example and video, as shown in Figure 10, 11.

Task Interface Details Each HIT contains a 1-over video and 6 rows, each corresponding to a legal delivery that occurred in the video. Each row consists of a dropdown for inputting the number of runs scored in that delivery, a checkbox for indicating an out ball occurred, a checkbox for indicating a wide ball occurred, and a field for writing the timestamp at which this information can be found. Figure 9 shows what the annotators initially see, as well as an example of how to fill it out.

LCric Annotation Verification A total of 205 overs with 1230 events spanning $\sim$1000 minutes were labeled by human annotators and compared to ground truth annotations from ESPNCricinfo. For each ball, we consider an event annotation to be correct if it was classified completely correctly. The timestamp annotation is marked as correct if it occurred anytime within the timestamp range specified by the ground truth ± 1 seconds.

LCric Annotation Statistics We found that in total, $1185/1230(96.34\%)$ of balls were classified correctly, while $1213/1230(98.62\%)$ of ball timestamps were marked correctly. Additionally, assuming human annotators can aggregate and reason easily with logic, we aggregate their annotations to answer queries in our test set, which provides our human baseline. We find that the human annotations achieve an accuracy of $5541/5740(96.53\%)$ on the test query set – exceeding the TQN and MemViT baselines by a large margin.

Queries	**GT probability**
atmost 7 1's	0.451
atleast 4 4's	0.523
atleast 5 1's AND atleast 3 4's	0.528
atleast 2 2's AND atleast 3 4's	0.452
atleast 4 4's AND atmost 5 o's	0.452
atleast 4 4's AND atmost 3 5's	0.456
atleast 4 2's OR atmost 2 4's	0.539
atleast 4 3's OR atmost 3 4's	0.544
atleast 5 2's OR atleast 4 4's	0.526
atleast 3 2's OR atleast 2 w's	0.485
atmost 3 4's AND atmost 2 6's	0.529
atmost 3 4's AND atmost 3 7's	0.544
atmost 2 0's OR atmost 3 4's	0.544
2 inrange [1, 6] AND 4 inrange [1, 4]	0.539
4 inrange [1, 6] AND o inrange [1, 4]	0.555
1 inrange [2, 7] OR 2 inrange [4, 5]	0.506
1 inrange [1, 2] OR 2 inrange [2, 3]	0.458
atleast 2 1's AND atleast 2 2's AND atleast 2 4's	0.542
atleast 4 4's OR atleast 4 o's OR atleast 4 w's	0.493
atleast 5 2's OR atleast 4 4's OR atleast 3 6's	0.535
atmost 4 3's AND atmost 3 4's AND atmost 2 5's	0.544
atmost 4 2's AND atleast 3 4's AND atmost 4 w's	0.546
atmost 5 1's OR atleast 5 3's OR atmost 2 4's	0.504
atmost 3 0's OR atleast 5 3's OR atmost 3 4's	0.544
atmost 3 0's OR atmost 4 1's OR atmost 2 4's	0.472
atmost 2 0's OR atmost 5 1's OR atmost 2 4's	0.504
1 inrange [2, 6] OR 2 inrange [3, 4] OR 3 inrange [6, 7]	0.528
atleast 4 0's AND atleast 3 1's AND atleast 2 2's AND atleast 2 4's	0.52
atleast 4 4's OR atleast 2 5's OR atleast 2 6's OR atleast 4 o's	0.518
atmost 3 2's AND atmost 4 4's AND atmost 3 6's AND atmost 5 w's	0.539
6 inrange [1, 7] OR 8 inrange [2, 4] OR o inrange [2, 3] OR w inrange [6, 7]	0.494
1 inrange [1, 6] OR 5 inrange [1, 2] OR o inrange [3, 6] OR w inrange [4, 6]	0.511

Table 4: The binary choice query set used for 10 over experiments and their associated ground truth (GT) probability of occurrence in the LCric train set.

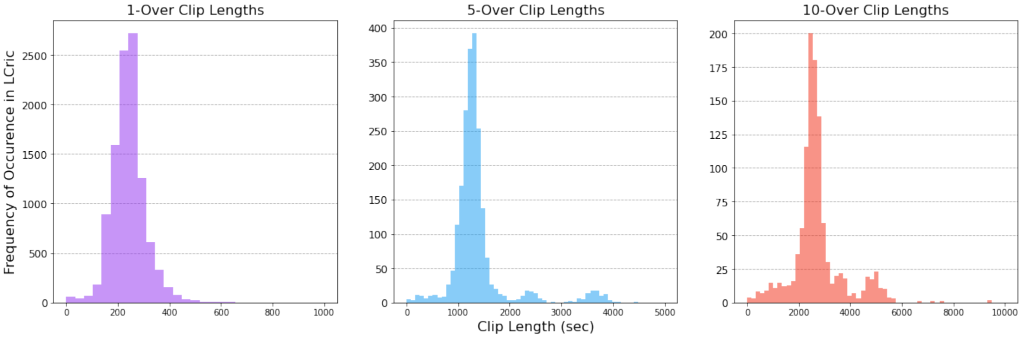


Figure 7: **Occurrence frequency of various clip lengths**. Here we plot the distribution of occurrences of various clips lengths (in seconds) that were extracted for LCric.

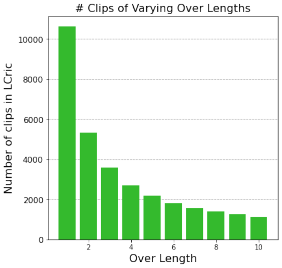
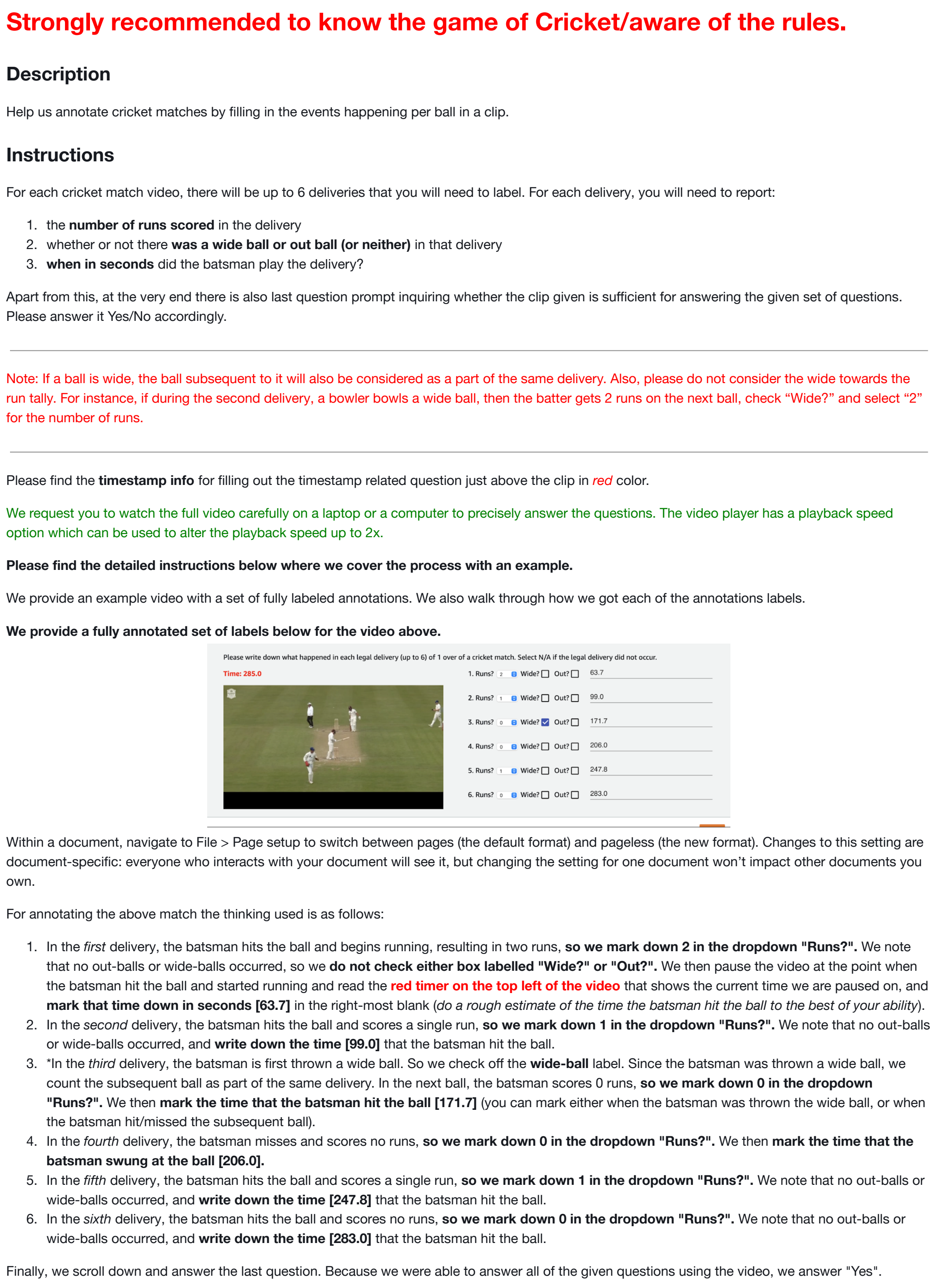


Figure 8: **Number of clips for different overs.** We plot the distribution of the number of clips for different over lengths in LCric.



Strongly recommended to know the game of Cricket/aware of the rules.

Description

Help us annotate cricket matches by filling in the events happening per ball in a clip.

Instructions

For each cricket match video, there will be up to 6 deliveries that you will need to label. For each delivery, you will need to report:

1. the **number of runs scored** in the delivery
2. whether or not there **was a wide ball or out ball (or neither)** in that delivery
3. **when in seconds** did the batsman play the delivery?

Apart from this, at the very end there is also last question prompt inquiring whether the clip given is sufficient for answering the given set of questions. Please answer it Yes/No accordingly.

Note: If a ball is wide, the ball subsequent to it will also be considered as a part of the same delivery. Also, please do not consider the wide towards the run tally. For instance, if during the second delivery, a bowler bowls a wide ball, then the batter gets 2 runs on the next ball, check “Wide?” and select “2” for the number of runs.

Please find the **timestamp info** for filling out the timestamp related question just above the clip in *red* color.

We request you to watch the full video carefully on a laptop or a computer to precisely answer the questions. The video player has a playback speed option which can be used to alter the playback speed up to 2x.

Please find the detailed instructions below where we cover the process with an example.

We provide an example video with a set of fully labeled annotations. We also walk through how we got each of the annotations labels.

We provide a fully annotated set of labels below for the video above.

Please write down what happened in each legal delivery (up to 6) of 1 over of a cricket match. Select N/A if the legal delivery did not occur.

Time: 285.0

1. Runs? 2 Wide? □ Out? □ 63.7
2. Runs? 1 Wide? □ Out? □ 99.0
3. Runs? 0 Wide? ☑ Out? □ 171.7
4. Runs? 0 Wide? □ Out? □ 206.0
5. Runs? 1 Wide? □ Out? □ 247.8
6. Runs? 0 Wide? □ Out? □ 283.0

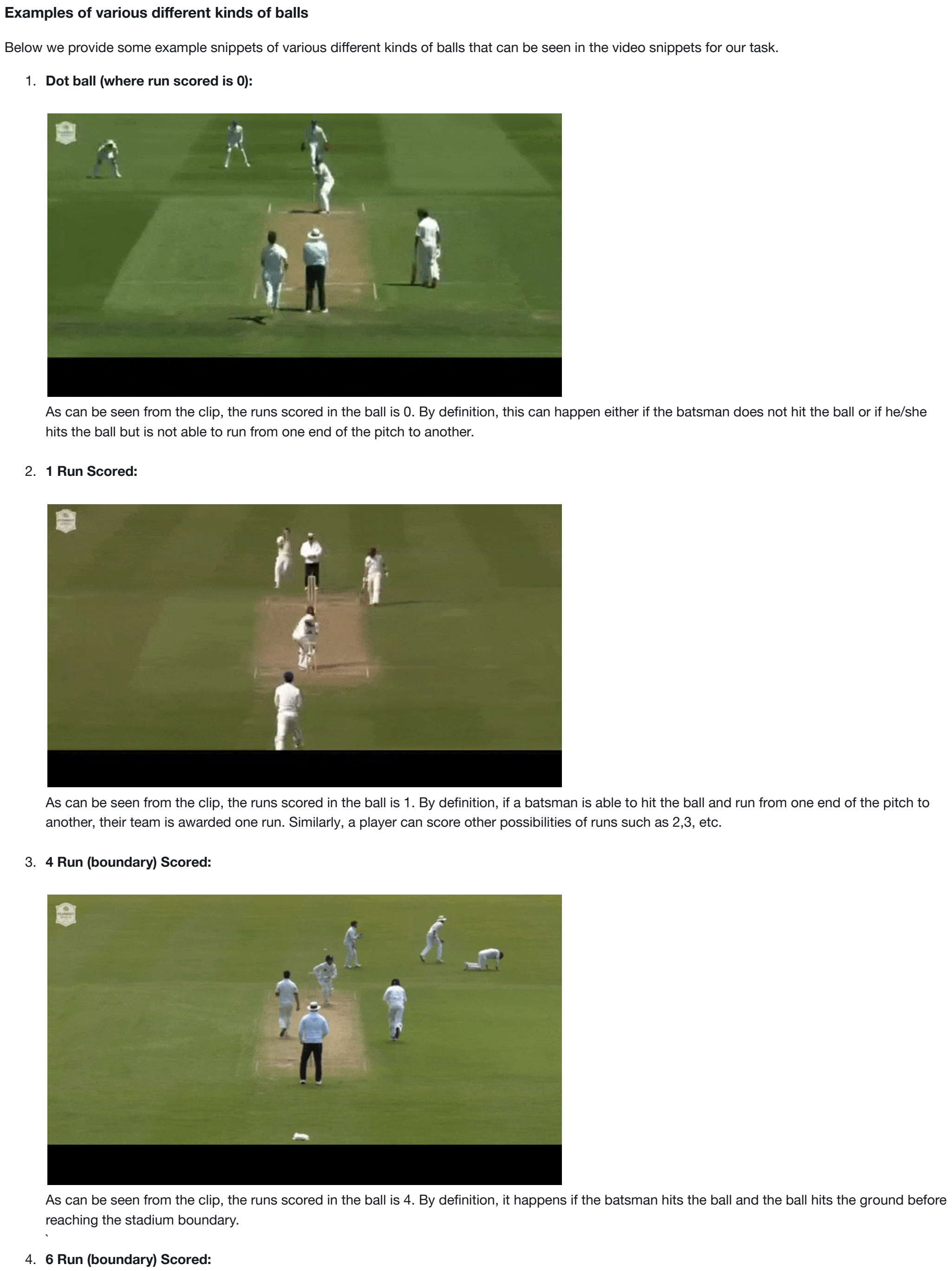
Within a document, navigate to File > Page setup to switch between pages (the default format) and pageless (the new format). Changes to this setting are document-specific: everyone who interacts with your document will see it, but changing the setting for one document won’t impact other documents you own.

For annotating the above match the thinking used is as follows:

1. In the *first* delivery, the batsman hits the ball and begins running, resulting in two runs, **so we mark down 2 in the dropdown "Runs?".** We note that no out-balls or wide-balls occurred, so we **do not check either box labelled "Wide?" or "Out?".** We then pause the video at the point when the batsman hit the ball and started running and read the **red timer on the top left of the video** that shows the current time we are paused on, and **mark that time down in seconds [63.7]** in the right-most blank (*do a rough estimate of the time the batsman hit the ball to the best of your ability*).
2. In the *second* delivery, the batsman hits the ball and scores a single run, **so we mark down 1 in the dropdown "Runs?".** We note that no out-balls or wide-balls occurred, and **write down the time [99.0]** that the batsman hit the ball.
3. *In the *third* delivery, the batsman is first thrown a wide ball. So we check off the **wide-ball** label. Since the batsman was thrown a wide ball, we count the subsequent ball as part of the same delivery. In the next ball, the batsman scores 0 runs, **so we mark down 0 in the dropdown "Runs?".** We then **mark the time that the batsman hit the ball [171.7]** (you can mark either when the batsman was thrown the wide ball, or when the batsman hit/missed the subsequent ball).
4. In the *fourth* delivery, the batsman misses and scores no runs, **so we mark down 0 in the dropdown "Runs?".** We then **mark the time that the batsman swung at the ball [206.0].**
5. In the *fifth* delivery, the batsman hits the ball and scores a single run, **so we mark down 1 in the dropdown "Runs?".** We note that no out-balls or wide-balls occurred, and **write down the time [247.8]** that the batsman hit the ball.
6. In the *sixth* delivery, the batsman hits the ball and scores no runs, **so we mark down 0 in the dropdown "Runs?".** We note that no out-balls or wide-balls occurred, and **write down the time [283.0]** that the batsman hit the ball.

Finally, we scroll down and answer the last question. Because we were able to answer all of the given questions using the video, we answer "Yes".

Figure 9: AMT instructions page given to annotators prior to starting the task.



Examples of various different kinds of balls

Below we provide some example snippets of various different kinds of balls that can be seen in the video snippets for our task.

1. **Dot ball (where run scored is 0):**

   As can be seen from the clip, the runs scored in the ball is 0. By definition, this can happen either if the batsman does not hit the ball or if he/she hits the ball but is not able to run from one end of the pitch to another.

2. **1 Run Scored:**

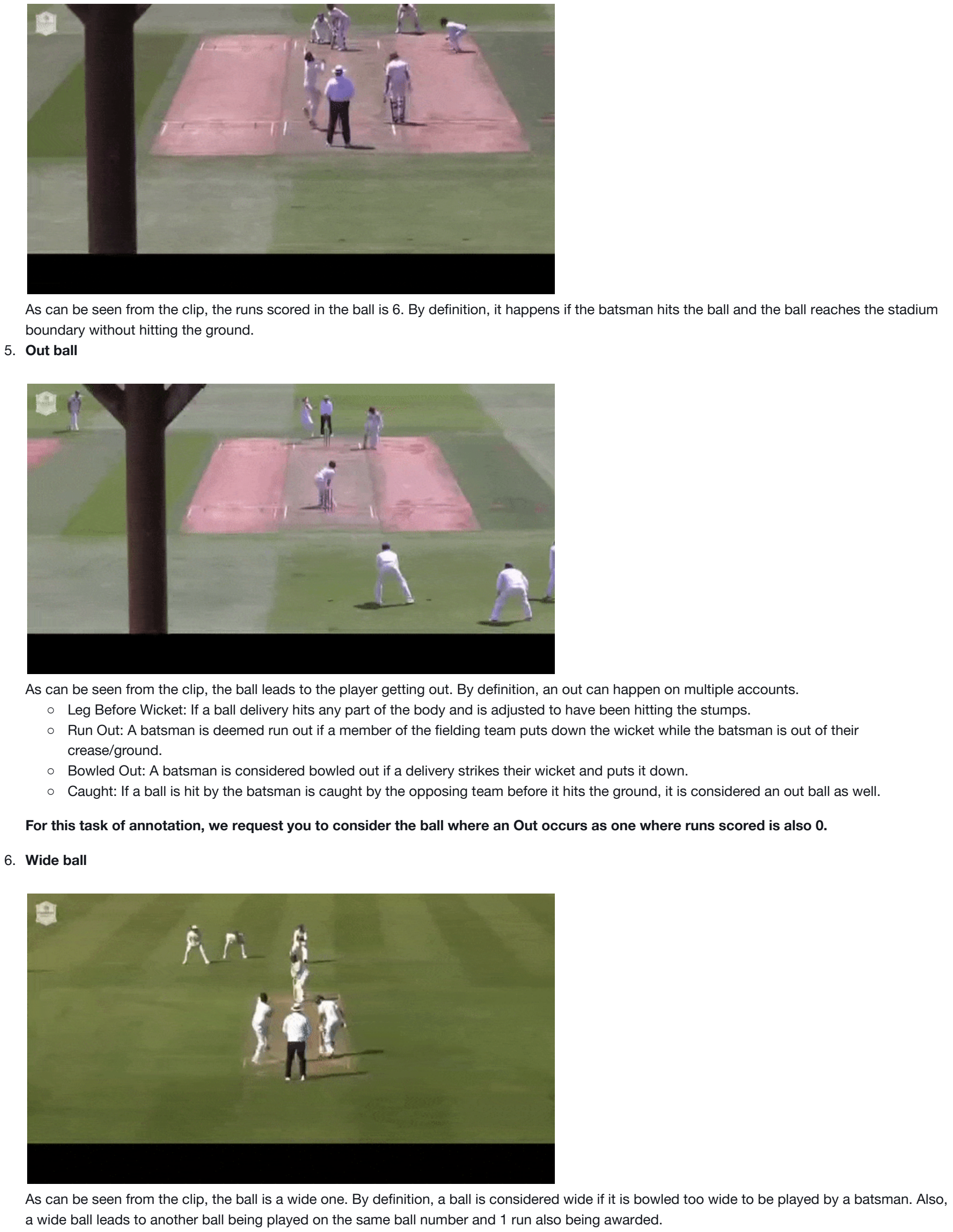
   As can be seen from the clip, the runs scored in the ball is 1. By definition, if a batsman is able to hit the ball and run from one end of the pitch to another, their team is awarded one run. Similarly, a player can score other possibilities of runs such as 2,3, etc.

3. **4 Run (boundary) Scored:**

   As can be seen from the clip, the runs scored in the ball is 4. By definition, it happens if the batsman hits the ball and the ball hits the ground before reaching the stadium boundary.
   `

4. **6 Run (boundary) Scored:**

Figure 10: Instructions page for AMT interface for Cricket. Each of the 12 events is described in gif format.



As can be seen from the clip, the runs scored in the ball is 6. By definition, it happens if the batsman hits the ball and the ball reaches the stadium boundary without hitting the ground.

5. **Out ball**

As can be seen from the clip, the ball leads to the player getting out. By definition, an out can happen on multiple accounts.

- Leg Before Wicket: If a ball delivery hits any part of the body and is adjusted to have been hitting the stumps.
- Run Out: A batsman is deemed run out if a member of the fielding team puts down the wicket while the batsman is out of their crease/ground.
- Bowled Out: A batsman is considered bowled out if a delivery strikes their wicket and puts it down.
- Caught: If a ball is hit by the batsman is caught by the opposing team before it hits the ground, it is considered an out ball as well.

For this task of annotation, we request you to consider the ball where an Out occurs as one where runs scored is also 0.

6. **Wide ball**

As can be seen from the clip, the ball is a wide one. By definition, a ball is considered wide if it is bowled too wide to be played by a batsman. Also, a wide ball leads to another ball being played on the same ball number and 1 run also being awarded.

Figure 11: Instructions page for AMT interface for Cricket. Each of the 12 events is described in gif format.

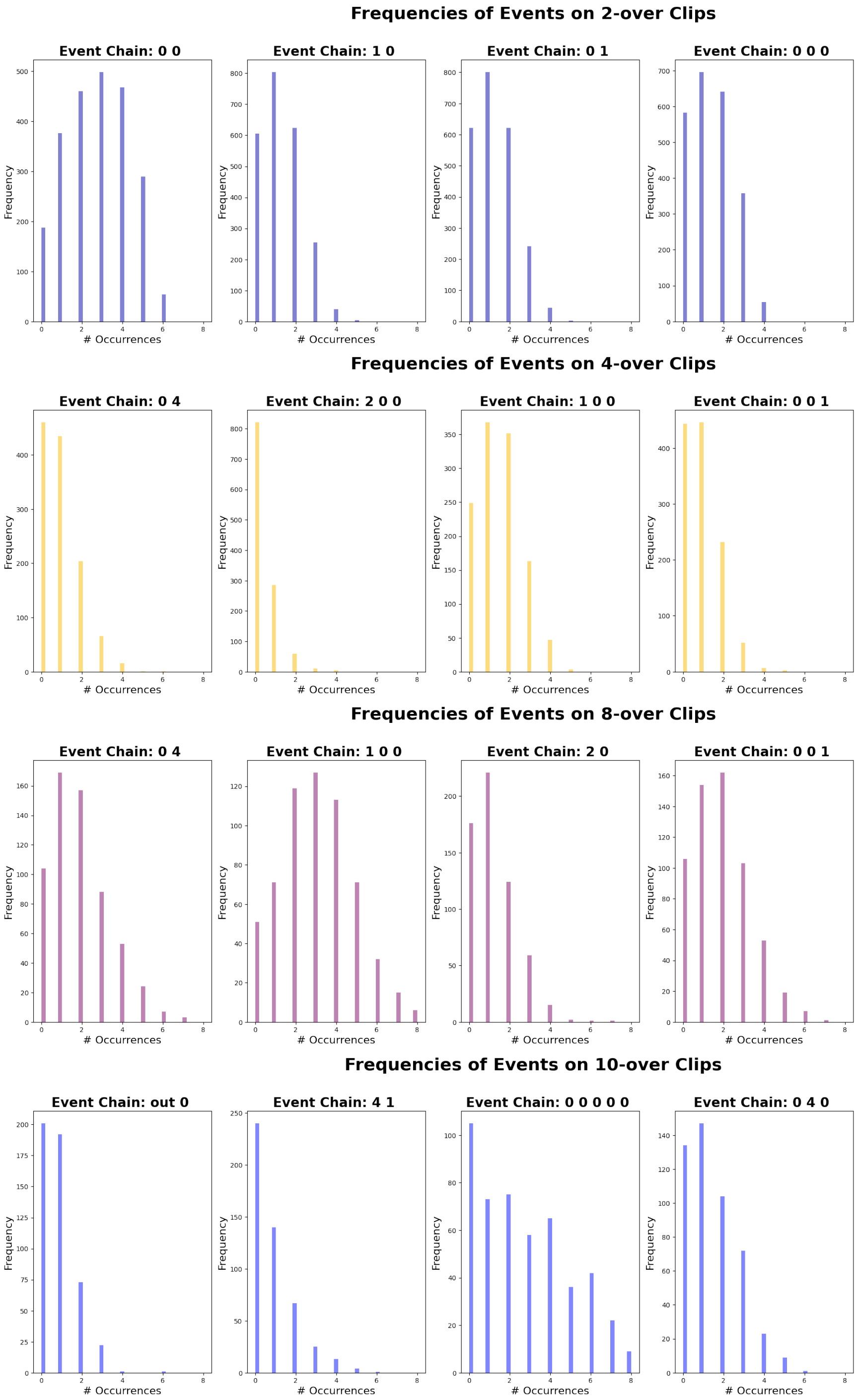


Figure 12: Ground truth output frequencies to some of the queries used in multi-choice query type questions in the train set of LCric for 2-over, 4-over, 8-over and 10-over clips.

Appendix C. Basketball Dataset (LBasketball)

We use the ASAP pipeline for annotating a new dataset, named LBasketball comprising of basketball dataset. The overall pipeline for the same is similar to one we utilized for creating LCric and mentioned in detail in Section 3.

The atomic events utilized for this are as follows: *foul*, *shot point* and *shot miss*. We annotate these events based on the description provided by ESPNInfo. The “shot miss” refers to when a player attempts to score by shooting the ball, but the ball misses the basket. The “shot point” refers to when a player scores by shooting the ball. A “foul” refers to any type of foul that a player makes, and it is clear by watching the referee and the players that a foul has occurred.

C.1 Statistics and Dataset Splits

Statistics. LBasketball currently includes $80+$ hours of basketball videos across 50 unique matches (average length of 1.5 hours). For processing in video understanding models we create clips of 2-3 mins average length comprising of 6 consecutive atomic events. All the videos are preprocessed at a resolution of 360p and we provide links to the source videos of higher resolution.

Dataset splits To maintain similar test setup with LCric, we split all matches in LBasketball into train, validation, and test splits and ensure a 3:1:1 ratio of the number of hours in each split.

C.2 LBasketball Queries

Similar to LCric, we also form compositional queries for LBasketball and form three different query sets, namely binary, multiple choice and regression queries (in a similar fashion as mentioned in Section 4.2). Some of the example queries we utilized for forming this set are as follows:
[Binary Choice Question] *Did a foul occur at-least 2 times AND Did a shot point occur at-most 3 times.* Answer: Yes/No
[Multiple Choice Question] *How many times did a foul occur just after a shot miss* Answer Option: 0, 1, 2, 4
[Regression Question] *How many points were overall scored in this video segment* Answer: 10
We also balance the queries just like we did in LCric and curate a set of 20 queries for binary-choice questions and 6 queries for multiple-choice questions.

#	Model	Binary Accuracy ↑	Multiple Accuracy ↑	Regression L1 Norm ↓
1	TQN	57.68%	19.05%	**17.21**
2	MovieChat	**60.21%**	**20.31%**	28.34
3	Human	96.34 %	96.29%	0.215

Table 5: Additional Results on the LCric dataset for 10-over clips. MovieChat outperforms TQN on binary and multi-choice query types while underperforming on the regression query.

Appendix D. Additional Results

Here, we outline some more results on LCric and also on our newly created basketball dataset. We specifically utilize 10 over clips for our LCric experiments and also have the same experimental setup as mentioned in section 5 in the main paper. For basketball dataset, we process clips at 2 fps (because of shorter clips lengths compared to cricket). We remove the scorecard from all frames to prevent annotation leakage and process frames at a resolution of 128 x 128. We compute following evaluation metrics: **1)** Classification accuracy for binary (*Binary Accuracy*) and multi-choice (*Multiple Accuracy*) queries *2)* Average L1 norm for regression queries (*Regression L1 Norm*)

We also use a new baseline, namely **MovieChat Song et al. (2023)**. The paper utilizes a memory mechanism inspired by Atkinson-Shiffrin memory model, and develop a long-form and short-form memory. Additionally, they also utilize ways to reduce the visual tokens and use a sliding windows approach to efficiently process the video. For querying, an LLM model is utilized on top of this to present a coherent answer taking into account the query and the video segment in question. This framework helps to efficiently query on top of really long videos.

D.1 Key Results

New Baseline for LCric Table 5 shows the performance of the new baseline MovieChat Song et al. (2023) and a comparison with the already present TQN and human baseline. While we clearly see a substantial improvement in binary and multi-choice questions using MovieChat, there is still considerable gap when compared to human performance. This improvement can clearly be attributed to better memory storing mechanism leading to better understanding over longer videos. We also see a dip in the regression

#	Model	Binary Accuracy ↑	Multiple Accuracy ↑	Regression L1 Norm ↓
1	TQN	69.43%	25.88%	**5.39**
2	MovieChat	**75.12%**	**26.83%**	8.89
3	Human	97.15 %	97.67%	0.582

Table 6: Results on Basketball dataset. Even with shorter clip lengths, we still see substantial performance gap between the video understanding baselines and the human evaluation.

#	Model	Binary Accuracy ↑	Multiple Accuracy ↑	Regression L1 Norm ↓
1	1 fps, 128*128	64.87%	21.56%	**14.74**
2	1 fps, 192*192	**64.93%**	**21.68%**	14.75
3	0.5 fps, 128*128	62.19%	20.52%	14.83

Table 7: Additional ablations on LCric dataset using TQN model for 2-over clips. We see improvements when using higher FPS and some minor gain when increasing the frame size as well.

performance. We believe this can be further improved a bit using better prompts and also other hyperparameter tuning of the model. Due to the limited time, we were not able to run further experiments ablating on these.

Results on LBasketball Table 6 shows the two baselines being run on the newly curated LBasketball. We also ran a small scale human study for our new basketball dataset as seen in the table. We see higher numbers overall for both the baselines as compared to LCric which can be attributed to the shorter clip lengths (2-3 mins) in LBasketball in comparison to the longer clips (50 mins) we used for LCric. Similar trends of MovieChat performing better can also be seen in our LBasketball evaluation as well.

Changing FPS and frame size Table 7 shows some results when we modify frame size and fps from our base hyperparams (0.5 FPS, 128*128). We see a clear improvement when increasing the frame rate, and also a slight improvement when using a better frame size. This was expected considering this leads to better localization of events. This further proves that better caching mechanisms and memory modules for more efficient computations are key for long-form video understanding.

Appendix E. LCric Datasheet

E.1 Motivation For Datasheet Creation

Datasheet as described in Gebru et al. (2021). Some questions were re-ordered to different sections, but the content remains the same.

Why was the datasheet created? (e.g., was there a specific task in mind? was there a specific gap that needed to be filled?)

Long-horizon Video Understanding (LVU) is the problem of reasoning over a long stream of video data, such as understanding the plot of a movie or analyzing the performance of a player in a lengthy game. Progress toward LVU has been greatly limited by the lack of dmensely annotated data. Creating an LVU benchmark requires manually annotating videos frame-by-frame, which is incredibly tedious and hard to scale. This constraint has limited the length of existing densely-annotated video understanding benchmarks from a few seconds to a few minutes. Thus, we created LCric, a dataset containing over 131 publicly available cricket matches (on YouTube) with 1000+ hours of match footage aligned with publicly available web annotations. LCric serves as a simple yet unsolved benchmark for long-horizon video understanding that evaluates a model's ability to localize and aggregate contextual information in long videos.

Has the dataset been used already? If so, where are the results so others can compare (e.g., links to published papers)?

Yes, see the results in Section 4 of the main paper.

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

LCric can also be used for short-form video understanding and action recognition tasks as well, but the primary focus and value of the benchmark is for long-horizon video understanding.

Who funded the creation of this dataset?

The project was supported by University of Toronto, Princeton, and Digital Research Alliance of Canada in providing us with resources for data collection and experimentation.

Any other comment?

We have curated an additional dataset on 50 publicly available NBA basketball games on YouTube (totally 80+ hours of footage). The following discussions on ethics and impacts of LCric similarly apply to this basketball dataset.

E.2 Datasheet Composition

What are the instances?(that is, examples; e.g., documents, images, people, countries) Are there multiple types of instances? (e.g., movies, users, ratings; people, interactions between them; nodes, edges)

The instances are videos of professional sports matches (primarily cricket) that were once broadcasted on TV for entertainment purposes and are now available on YouTube for public viewing. They primarily contain footage of professional, contracted athletes playing sports. While these matches sometimes contain cameras panning to audience members, our annotation pipeline used to collect these videos, ASAP, filters out frames that do not show the sports match. Furthermore, in all of the cricket matches used, there are no visible people other than the players on the field.

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

LCric consists of 131 unique professional cricket matches, totally over 1000 hours of raw match footage. All matches are publicly available on YouTube.

What data does each instance consist of? "Raw" data (e.g., unprocessed text or images)? Features/attributes? Is there a label/target associated with instances? If the instances related to people, are subpopulations identified (e.g., by age, gender, etc.) and what is their distribution?

Each instance consists of several hours of continuous footage from a professional sports match. Furthermore, there is an associated set of annotations stored as a JSON that contains a range of timestamps, as well as corresponding labelled events for these timestamps. Each event corresponds to an action in the game. For Cricket, this is specifically the batting team hitting the ball for a certain number of runs, hitting out, or hitting a ball. Finally, all sports matches are of male athletes, but in the instance of a video understanding benchmark where the labels correspond to in-game actions, we believe that this distribution is not harmful.

Is there a label or target associated with each instance? If so, please provide a description.

Each frame of LCric is labelled with an associated event, as described in the section above. The event is generally a text description of some general action that occurred in the game.

Is any information missing from individual instances? If so, please provide a description, explaining why this information is missing (e.g., because it was unavailable). This does not include intentionally removed information, but might include, e.g., redacted text.

There are no spectators or non-players that can be seen in any LCric matches because there are none in the raw match footage. In the basketball dataset, any scenes focusing on audience members are filtered out by our ASAP pipeline because there is also no relevant game information in these scenes.

Are relationships between individual instances made explicit (e.g., users' movie ratings, social network links)? If so, please describe how these relationships are made explicit.

The teams playing in each match and the players on each team are made explicit by their jerseys, but this is all public information.

Does the dataset contain all possible instances or is it a sample (not necessarily random) of instances from a larger set? If the dataset is a sample, then what is the larger set? Is the sample representative of the larger set (e.g., geographic coverage)? If so, please describe how this representativeness was validated/verified. If it is not representative of the larger set, please describe why not (e.g., to cover a more diverse range of instances, because instances were withheld or unavailable).

This dataset represents a subset of all publicly available professional cricket matches on the web. However, we believe that the choice of which matches are included in the dataset would not affect the reasoning ability of a video understanding model, and is therefore representative of the larger set.

Are there recommended data splits (e.g., training, development/validation, testing)? If so, please provide a description of these splits, explaining the rationale behind them.

The recommended data splits used in our baseline experiments were 60% of the matches for train, 20% for validation, and the remaining 20% for test.

Are there any errors, sources of noise, or redundancies in the dataset? If so, please provide a description.

Because we aligned web annotations to each video using an automatic annotation pipeline (ASAP), there may exist errors in the labelling of certain events to certain frames. However, we have also conducted a crowdsourced study on the accuracy of our annotation pipeline on LCric.

Is the dataset self-contained, or does it link to or otherwise rely on external resources (e.g., websites, tweets, other datasets)? If it links to or relies on external resources, a) are there guarantees that they will exist, and remain constant, over time; b) are there official archival versions of the complete dataset (i.e., including the external resources as they existed at the time the dataset was created); c) are there any restrictions (e.g., licenses, fees) associated with any of the external resources that might apply to a future user? Please provide descriptions of all external resources and any restrictions associated with them, as well as links or other access points, as appropriate.

The videos we collected for LCric are from publicly available videos on YouTube, generally from the official source. Furthermore, the web annotations for almost all professional cricket matches are available publicly on the website ESPNCricinfo.com. We currently have scraped versions of the videos with links to the original, but if the original were to be removed for whatever reason, we would remove the corresponding video. Furthermore, the benefit of our ASAP pipeline is that it becomes easier to align newly available matches/videos for use in training.

Any other comments?

N/A

E.3 Collection Process

What mechanisms or procedures were used to collect the data (e.g., hardware apparatus or sensor, manual human curation, software program, software API)? How were these mechanisms or procedures validated?

Both the cricket matches and web commentary/annotation data were scraped using a video downloader or a simple scraping software. The scraping was verified manually.

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)? If data was reported by subjects or indirectly inferred/derived from other data, was the data validated/verified? If so, please describe how.

The data associated with each instance was first scraped from ESPNCricinfo.com, then aligned frame-by-frame with each video using the ASAP pipeline. The data is directly observable on the web.

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

Videos were chosen by searching for publicly available cricket matches on YouTube, preferably from official sources. Furthermore, these matches had to have associated play-by-play match statistics available on ESPNCricinfo.

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

We paid crowdworkers through Amazon Mechanical Turk (AMT) to verify a randomly sampled subset of clips of our annotated Cricket matches. Each clip was roughly 5 minutes and could be answered instantly upon viewing the clip, and we paid each crowdworker $1.50 per clip. This roughly equates to $18 per hour, which is well above the minimum wage in any country in the world (although we limited our crowdsourcing to India and New Zealand, where cricket is more popular).

Over what timeframe was the data collected? Does this timeframe match the creation timeframe of the data associated with the instances (e.g., recent crawl of old news articles)? If not, please describe the timeframe in which the data associated with the instances was created.

The data was collected over the last two years.

E.4 Data Preprocessing

Was any preprocessing/cleaning/labeling of the data done (e.g., discretization or bucketing, tokenization, part-of-speech tagging, SIFT feature extraction, removal of instances, processing of missing values)? If so, please provide a description. If not, you may skip the remainder of the questions in this section.

Removal of non-sport elements of each video was done automatically with ASAP. Essentially, if there was no scorecard information on the screen, then the frames were removed by ASAP, as they are not relevant to the video understanding benchmark.

Was the "raw" data saved in addition to the preprocessed/cleaned/labeled data (e.g., to support unanticipated future uses)? If so, please provide a link or other access point to the "raw" data.

The raw data is saved on YouTube, as all matches are publicly available for viewing. We have included links to each corresponding video that was used.

Is the software used to preprocess/clean/label the instances available? If so, please provide a link or other access point.

The software used, ASAP, is described in the main paper. It will be made available with the release of the full paper.

Does this dataset collection/processing procedure achieve the motivation for creating the dataset stated in the first section of this datasheet? If not, what are the limitations?

This dataset collection process achieves the motivation of creation a long-horizon video benchmark described in the first section. However, there are some limitations of ASAP that we would like to address. Firstly, ASAP requires free, publicly available videos and web annotation data with some kind of identifier (e.g. match state) that can be aligned to the videos. Thus, sports videos are the ideal candidate, but it is therefore difficult to apply ASAP to other domains of videos. Secondly, ASAP is an automatic pipeline for long videos that relies on pre-existing annotations, and hence it relies on the accuracy of these annotations. Furthermore, while we can verify the accuracy of ASAP using crowdworkers, we cannot afford to do this for 1000+ hours of footage.

We should point out that our goal in this work was to present a preliminary long-horizon video benchmark for video understanding models that should be quite simple to solve if the video understanding model exhibits basic localization and aggregation abilities. In other words, a video understanding model that can solve a more complex video understanding benchmark should also be able to solve our long-horizon sports benchmark. Thus, we believe that extending ASAP to other scenes/domains becomes important after video understanding baselines can solve the simple sports benchmark.

Any other comments

N/A

E.5 Dataset Distribution

How will the dataset be distributed? (e.g., tarball on website, API, GitHub; does the data have a DOI and is it archived redundantly?)

We have a list of all of the relevant YouTube videos and their links, as well as a Python script for downloading and using the videos in GitHub. This way, if a video is taken down in the future, the script will be unable to use it, preventing the distribution of non-publicly available material. Furthermore, the script can be easily updated to include other publicly available sports matches. Also, we have provided a Google Drive link with the scraped and frame-aligned annotations as well. This link is provided in the same GitHub repository.

When will the dataset be released/first distributed? What license (if any) is it distributed under?

The dataset will be released as soon as this work is published. The dataset as a whole and the frame-aligned annotations will be released under the ODC-By 1.0 license. Individual videos are subject to the licenses by the owners of these videos (sports organizations), and users need to assess these licenses based on downstream use cases.

Are there any copyrights on the data?

The frame-aligned annotation data is not copyrighted, but the videos are. This is why we provide the annotation data, but not the YouTube videos, and only offer a downloader for use of the videos in training and inference. We do not own the rights to the professional sports matches.

Are there any fees or access/export restrictions?

There are no fees. All data is publicly available on both YouTube and ESPNCricinfo, both of which are well supported. Users must follow the license of the content with which the original files were distributed and the terms of service for each platform.

Any other comments? N/A

E.6 Dataset Maintenance

Who is supporting/hosting/maintaining the dataset?

The authors of the dataset will maintain the dataset for the forseeable .

Will the dataset be updated? If so, how often and by whom?

The dataset will not be updated, unless there is a severe issue with regards to the availability of videos in the dataset.

How will updates be communicated? (e.g., mailing list, GitHub)

Updates will be communicated through our GitHub organization and repositories.

If the dataset becomes obsolete how will this be communicated?

It will be communicated through our GitHub organization and repositories.

Is there a repository to link to any/all papers/systems that use this dataset?

We may update our GitHub repository with a list of works using this dataset.

If others want to extend/augment/build on this dataset, is there a mechanism for them to do so? If so, is there a process for tracking/assessing the quality of those contributions. What is the process for communicating/distributing these contributions to users?

Yes, the purpose of the ASAP pipeline is to be able to easily gather more annotated data for the long-horizon video understanding task.

E.7 Legal and Ethical Considerations

Were any ethical review processes conducted (e.g., by an institutional review board)? If so, please provide a description of these review processes, including the outcomes, as well as a link or other access point to any supporting documentation.

Institutional review boards were not involved in the collection of the dataset.

Does the dataset contain data that might be considered confidential (e.g., data that is protected by legal privilege or by doctorpatient confidentiality, data that includes the content of individuals non-public communications)? If so, please provide a description.

To our knowledge, this dataset does not contain any data that might be considered confidential.

Does the dataset contain data that, if viewed directly, might be offensive, insulting, threatening, or might otherwise cause anxiety? If so, please describe why

To our knowledge, this dataset does not contain any data that might be considered offensive or insulting.

Does the dataset relate to people? If not, you may skip the remaining questions in this section.

Yes, the dataset contains professional athletes playing their sport.

Does the dataset identify any subpopulations (e.g., by age, gender)? If so, please describe how these subpopulations are identified and provide a description of their respective distributions within the dataset.

All games present in the dataset are of male athletes. However, this is primarily due to the abundance of web annotations and publicly available games. Furthermore, this does not affect the annotations or benchmark in any way.

Is it possible to identify individuals (i.e., one or more natural persons), either directly or indirectly (i.e., in combination with other data) from the dataset? If so, please describe how.

In most instances, the players are too small to see detailed facial or body features. However, in some instances a player's jersey is clear enough to see their last name and team number.

Does the dataset contain data that might be considered sensitive in any way (e.g., data that reveals racial 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)? If so, please provide a description.

To our knowledge, this dataset does not contain any data that might be considered sensitive.

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

We collected the data from the sports organizations that broadcast these matches on YouTube.

Were the individuals in question notified about the data collection? If so, please describe (or show with screenshots or other information) how notice was provided, and provide a link or other access point to, or otherwise reproduce, the exact language of the notification itself.

No, individuals were not notified about the collection of the dataset.

Did the individuals in question consent to the collection and use of their data? If so, please describe (or show with screenshots or other information) how consent was requested and provided, and provide a link or other access point to, or otherwise reproduce, the exact language to which the individuals consented.

No, but professional athletes do consent to being recorded and broadcasted for their professional sports games. Furthermore, gathering statistics through publicly available videos is a common and legal practice in sports.

If consent was obtained, were the consenting individuals provided with a mechanism to revoke their consent in the future or for certain uses? If so, please provide a description, as well as a link or other access point to the mechanism (if appropriate).

N/A

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? If so, please provide a description of this analysis, including the outcomes, as well as a link or other access point to any supporting documentation.

No analysis has been conducted.

Any other comments?

Potential negative societal impacts. We acknowledge that there are few potential negative societal impacts of ASAP and LCric. Firstly, the availability of a tool for aligning web annotations to videos may cause misuse of copyrighted and/or private property. Secondly, our use of online sports videos, which are copyrighted material, may lead to misuse and illegal distribution of these materials. Finally, while the limitations on the type of data that may be aligned by ASAP prevents the extension of the tool to potentially harmful and invasive applications, we acknowledge that it may be possible in some instances for ASAP to extend harmful forms of video footage such as surveillance.