Appendix

Table of Contents

Supplementary video is here: https://drive.google.com/file/d/1zoYA2As2igeVghdZPmfOnKxmgXFuJUyG/view?usp=sharing

A. Data Collection

This section overviews the collection procedures and scenarios per site.

International Institute of Information Technology (IIIT), Hyderabad, India: At IIIT, Hyderabad, we followed a protocol of distributed data collection with a centralized team doing coordination and verification. We first identified local coordinators in different parts of the country and explained the data collection plans, goals and process. They then helped in collecting data in their own local regions from natural settings with informed participants. Participants were recruited locally considering the range of activities, and also the guidelines and restrictions of COVID-19. The central team could not travel to all these locations for training the coordinators or collecting the data. We shipped multiple cameras to the local coordinators and remotely guided them on data collection following the COVID protocols. The collected data and consent forms were then shipped back to the university, where manual verification, de-identification (wherever applicable), and sharing with the consortium took place.

At IIIT Hyderabad, we recorded 660.5 hours of data with the help of 138 subjects. The videos were collected in 5 different states in India, geographically well apart. We cover 36 different scenarios, such as making bricks using hands, knitting, making egg cartons, and hairstyling. The age of subjects ranged from 18-84 years with 10 distinct professional backgrounds (teachers, students, farmers, blacksmiths, homemakers, etc.). Out of all the subjects, 94 were males, and 44 were females. We use GoPro Hero 6 and GoPro Hero 7 for recording the videos. The GoPro's were shipped to the participants in different parts of the country. Videos were shared back either in external hard disks or over the cloud storage. Each video was manually inspected for any sensitive content before sharing.

Primary contributors: Raghava Modhugu - data collection pipeline, design of the setup and workflow. Siddhant Bansal - IRB application, consent forms and de-identification. C. V. Jawahar - lead contributor for data collection. We also acknowledge the contributions of Aradhana Vinod (coordination and communication), Ram Sharma (local data management and verification), and Varun Bhargavan (systems and resources).

University of Tokyo, Japan: We recruited 81 Japanese participants (41 male, 40 female) living around Tokyo, Japan through a temporary employment agency. The participant's gender and age (from the 20s to 60s) were balanced to collect diverse behavior patterns. We focused on two single-actor activities: cooking (40 participants, 90 hours) and handcraft (41 participants, 51 hours). In the cooking scenario, participants were asked to record unscripted videos of cooking at their homes. In the handcraft scenario, participants visited our laboratory and performed various handcraft activities (e.g., origami, woodworking, plastic model, cutout picture). We collected data using GoPro HERO 7 Black camera for cooking and Weeview SID 3D stereo camera for handcraft. Our data collection protocol was reviewed and approved by University of Tokyo ethical review board.

Primary contributors: Yoichi Sato – lead coordinator for data collection, Takuma Yagi and Takumi Nishiyasu – contributed to participant recruiting, protocol design, data collection and inspection, and IRB submission, Yifei Huang and Zhenqiang Li – contributed to data inspection and transfer, Yusuke Sugano – contributed to selecting video recording scenarios, protocol design and IRB submission.

University of Bristol, UK: Participants were recruited through adverts on social media and university internal communication channels. These participants then spread the word to their acquaintances and some participants joined the project through word-of-mouth recommendations of previous participants. Data was collected between Jan and Dec 2020, from 82 participants. With the pandemic taking over in March, the project shifted to online operation where cameras were posted, and training took place over Zoom meetings. Participants first expressed interest by sending an email and they were provided with an information sheet. This was followed by a preliminary Zoom meeting with a researcher to brief participants about the procedure, answer any questions and agree on the scenarios to be recorded.

We set a limit to the total number of minutes per scenario, to increase diversity of recordings. For example, driving cannot be longer than 30 minutes while cooking can be up to 1.5 hours. Each participant was instructed to record a minimum of 2 hours across 4 scenarios. Importantly, participants were encouraged to collect activities they naturally do. For example if one regularly cycles or practices music, they were asked to record these scenarios. Additionally, paired scenarios (people cooking together or playing games) were encouraged and multiple (2-3) cameras were posted for participants sharing a household. All participants signed a consent form before a camera was posted to their residence. Cameras were posted to 9 UK cities in England, Wales and Scotland including one participant in the Isle of North Uist.

Upon receipt of the camera, a second Zoom meeting was scheduled to train the participant on the equipment and detail how footage is reviewed and uploaded. Participants were given 2 weeks to record, with an additional week of extension upon request. Once recording is completed, footage is uploaded by the participant and reviewed for good lighting, correct setting and viewpoint. Participants were reimbursed for their participation in the project.

Scenarios recorded in the UK covered: commuting (driving, walking, cycling, taking the bus, hiking, jogging), entertainment (card games, board games, video games, lego, reading, practising a musical instrument, listening to music, watching TV), jobs (lab work, carpentry), sports (football, basketball, climbing, golf, yoga, workouts) and home-based daily activities (cooking, cleaning, laundry, painting, caring for pets, tidying, watering the plants), DIY (fixing, gardening, woodwork) and crafts (colouring, crafting, crochet, drawing, knitting, sewing). Footage was captured using GoPro Hero-7, Hero-8 and Vuzix.

Footage was then reviewed by researchers to identify any PII. 36% of all videos required de-identification. We used Primloc's Secure Redact software suite, with integrated tools and user interfaces for manual tracking and adjusting detections. Redacted recordings were reviewed manually, then encoded and uploaded to the AWS bucket. During encoding, IMU meta data was separately extracted. Integrated audio and video using native 50fps recordings are available.

In total, 262 hours were recorded by 82 participants. On average, each participant recorded 3.0 hours ($\sigma = 0.7$ hours) The data is published under General Data Protection Regulation (GDPR) compliance.

Primary contributors: Michael Wray - data collection, consent forms and information sheets; Jonathan Munro - data collection and ethics application; Adriano Fragomeni - data collection and de-identification oversight; Will Price - data ingestion, encoding and metadata; Dima Damen - scenarios, procedures, data collection oversight and participant communication. We acknowledge the efforts of Christianne Fernee in manually reviewing all data.

Georgia Tech, Atlanta, GA, USA: Participant groups from the Atlanta, Georgia, USA metro area were recruited via online posts and advertisements on sites such as Facebook, Reddit, and Instagram. Each group of participants was comprised of friends or family members who knew each other prior to participating in the study. Participants were required to be aged 18-64, to not be considered high risk for COVID-19, and to be able to play social deduction games in English. Our study protocol was reviewed and approved by the Georgia Tech Institutional Review Board (IRB). In total, approximately 43 hours of egocentric video were collected from 19 participants (per participant disclosure - 10 male, 7 female, 1 non-binary, 1 not reported). Participants had a mean age of 31.6 years with 7 participants aged 20-29 years, 10 participants aged 30-39 years, and 2 participants aged 40-49 years.

Participants wore an egocentric head-worn camera and on-ear binaural microphones. Some participants wore the ORDRO EP6 camera while others wore the Pupil Invisible cameras. The audio was recorded using a Tascam DR-22WL and Sound Professionals MS-EHB-2 Ear-hook binaural microphones. A third-person video was also captured via a Logitech C930e Webcam. Participants wore the provided recording devices while eating, drinking, and playing social deduction games such as *One Night Ultimate Werewolf* and *The Resistance: Avalon* in their own home. This at-home game-night setting elicited a wide range of spontaneous and naturalistic social behaviors and interactions. In addition, eating and drinking behaviors were captured from both the egocentric and third-person cameras.

In addition to participating in the recorded session, participants completed a survey that captured their demographic information. All data was screened and censored by study personnel to remove any identifying information including visible personal information on their phone screens or the exterior of the home. Participants also had the opportunity to review the videos and request additional censoring.

Primary contributors: Fiona Ryan - lead coordinator for data collection, including synchronization, de-identification, and ingestion; Audrey Southerland - lead coordinator for IRB development and recruiting; Miao Liu - contributed to data collection and ingestion; James M. Rehg - contributed to protocol design and data collection.

Indiana University, Bloomington, IN, USA: Participants in the Bloomington, Indiana, USA area were recruited through advertisements on social media, online classifieds boards, and email lists. We also used snowball sampling by asking participants to share our ads with their friends. We recruited participants who were willing to perform interactive small group activities such as playing sports, playing board or card games, playing musical instruments, assembling puzzles, etc. The health of participants and study personnel was safeguarded by collecting data either outdoors (where people can more safely interact without wearing masks), or indoors in the homes of the participants. In either case, we initially required that all participants in a social group be part of the same household to minimize the risk of spreading disease between households, but later we allowed groups of people who were comfortable interacting with one another (e.g., because they are vaccinated for COVID-19). Group sizes ranged from 1 to 6 people, with groups of 2 or 3 being the most common.

We collected data with four different devices: zShade 1080p camera glasses, iVue Rincon 1080 camera glasses, ORDRO EP-6, and Pupil Labs Invisible camera and gaze tracking glasses. We used multiple devices because each has various advantages and disadvantages; zShade has a large horizontal field of view, for example, while iVue has an adjustable vertical field of view, ORDRO sits by the ear and is mounted on a headband which works well for people wearing prescription glasses, and Invisible offers gaze tracking but is very expensive. We asked as many participants as possible in the group to wear cameras. We primarily used our two Pupil Labs Invisibles whenever possible, because of their ease of use and ability to collect gaze data, but we also used the ORDRO EP-6 when there were larger groups or when participants wore prescription glasses.

Our protocol was reviewed and approved by the Indiana University Institutional Review Board (IRB). We first conducted an online meeting with potential participants to describe the study, explain the use of the cameras, agree on an activity for them to perform, and answer their questions. We ask participants to try to limit capture of potentially privacy-sensitive content by choosing a place within their home that did not have personally identifiable information, by avoiding recording people other than those participating in the study, and by avoiding saying last names or other sensitive audio.

We then arrange a time to meet them, typically outside their

home or in an outdoor public place. We set up the cameras, help the participants put them on, give them our contact information in case they have any problems, and then we leave while they perform the activity. We then return after about one hour to pick up the cameras. Within a few days, we send each participant a copy of the video taken by their camera, and ask them to review the footage and identify any privacy-sensitive content (video or audio) that they would prefer to be blurred or removed. We manually edit out any such content (using Adobe Premiere Pro). We also review all video for faces of non-participants and personally-identifying information such as house numbers or license plates, and blurred these accordingly. We use Pupil Labs software to synchronize eye gaze with the video for each participant, and then used Adobe Premiere Pro to temporally synchronize video across different participants using audio track comparison.

In total, approximately 103 hours of video were collected from 66 participants (42 female, 23 male, 1 non-binary; for age, 46 were 20-29 years old, 14 were 30-39 years old, 1 was 40-49, 2 were 50-59, 1 was 60-69, and 2 were 70-79).

Primary contributors: David Crandall - lead coordinator for data collection; Yuchen Wang - contributed to protocol design, participant recruiting, and data collection; Weslie Khoo - developed multi-camera synchronization and de-identification pipelines.

University of Minnesota, Twin Cities, MN, USA: Participants in the Minneapolis and St. Paul, Minnesota, USA area were recruited through advertisements on social media and university bulletins such as Facebook AD, Craiglist, and Redhat. A total of approximately 313 hours of data was collected from 45 participants (22 males and 23 females). Age groups include 5 teenagers, 20 people in their twenties, 11 people in their thirties, 8 people in their forties, and 1 person in their fifties. We recruited participants as multiple groups and encouraged them to engage in unstructured natural social interactions. Such interactions included playing card games, talking in the kitchen while cooking, playing basketball, and building a tent at a camp site. In all cases, we required that all participants in a social group be part of the same household to minimize the COVID-19 risk. Group sizes ranged from 1 to 6 people, with groups of 2 or 3 being the most common.

We collected data with the zShade 1080p camera glasses that have a large field of view. Our protocol was reviewed and approved by the University of Minnesota Institutional Review Board (IRB). We first conducted an online meeting with potential participants to describe the study, explain the use of the cameras, agree on an activity for them to perform, and answer their questions. We then arranged a time for them to receive the cameras and provided them with a postage-paid box for camera return. A few days later, participants shipped the cameras to our designated return address. We downloaded the data after sanitizing cameras and equipment. After the data capture was complete, we visually inspected every second of video in order to exclude any privacy-sensitive information (e.g. license plates, smart phone screens, and credit card numbers), and to assess the duration of non-social activities. For incidental participants (i.e. bystanders) appearing in data collected by the camera wearer in public settings (e.g., shopping, concert, at a park, etc.), data collection consists only of recording publicly observable behavior with no manipulation or direct interaction with the participants, and this university's IRB allows an assumed waiver of consent for those participants.

Primary contributors: Hyun Soo Park - lead coordinator for data collection; Jayant Sharma - contributed to participant recruiting, data collection, IRB submission, analysis, and data ingestion.

National University of Singapore, Singapore: Participants were recruited from Singapore through advertisements on social media, via flyers and surveys, as well as from sourcing by the project coordinator. Residents of Singapore aged 21 to 70 who could wear a camera while participating in social sessions were eligible for inclusion in our study. During the recording session, the participants were required to attend social events such as family gatherings, exercising with a trainer, hairdressing, getting manicure, attending a session for teaching assistants, attending a group meeting, etc. The devices used for data collection were GoPro Hero 8, GoPro Hero 9, and AR glasses. GoPro cameras have binaural microphones while the AR glasses can only record mono audio. In total, 51 hours of videos were collected from 40 participants (25 males and 15 females). Age groups include 31 twenties, 5 thirties, 3 fifties, and 1 sixties.

Primary contributors: Mike Zheng Shou - lead coordinator for data collection; Eric Zhongcong Xu - contributed to data collection; Ruijie Tao - contributed to data collection.

Meta Reality Labs, Redmond, WA, USA: Participants were recruited from the Seattle area through a FRL-hired vendor company. In total, there were 400 hours collected from 206 unique participants in 6 scenes staged in FRL's research labs in 2019. The ethnic groups include 50.8% Caucasian, 28.2% African, 11.9% Asian and 9% Hispanic. The staged environments include four types of apartments, a clothing store, and a grocery store. During the recording sessions, the participants were asked to wear Vuzix glasses to go through the following everyday scenarios as naturally as possible: grocery shopping, buying clothes, watching TV, playing video games, listening to music, dancing, weight lifting, stretching, reading email, paying bills, online gaming, cooking, talking with other people, meetings, whiteboarding, and video calling. The emails and bills were always mock data, not personal emails or bills of the participants. The video calls took place between participants only.

Three out of four apartments have corresponding 3D scans. We use the state-of-the-art dense reconstruction system [209] to obtain the 3D photo-realistic reconstruction of those apartments. Volumetric representations are obtained from a customized capture rig and dense 3D meshes are extracted by the Marching Cubes algorithm with textures. We further annotate the dense meshes by labeling object categories over the mesh polygons; 35 object categories plus a background class label are used in annotation.

Primary contributors: Mingfei Yan, Richard Newcombe, Kiran Somasundaram, Chao Li.

Universidad de los Andes, Colombia: We gather 302.5 hours across 20 scenarios from 77 unique participants. We record videos using GoPro Hero 9 cameras between July and August 2021. We recruit volunteer participants from within the Uniandes community and their families and friends. The ethnic groups include 89.9% Hispanic, 1.4% African, and 5.8%Caucasian. The gender distribution follows 41.6% male and 58.4% female with ages ranging from 18 to 65 (6 teens, 44 twenties, 3 thirties, 2 forties, 6 fifties, and 1 sixties). Our data collection focuses mainly on simultaneous video recording in groups of camera wearers within a common

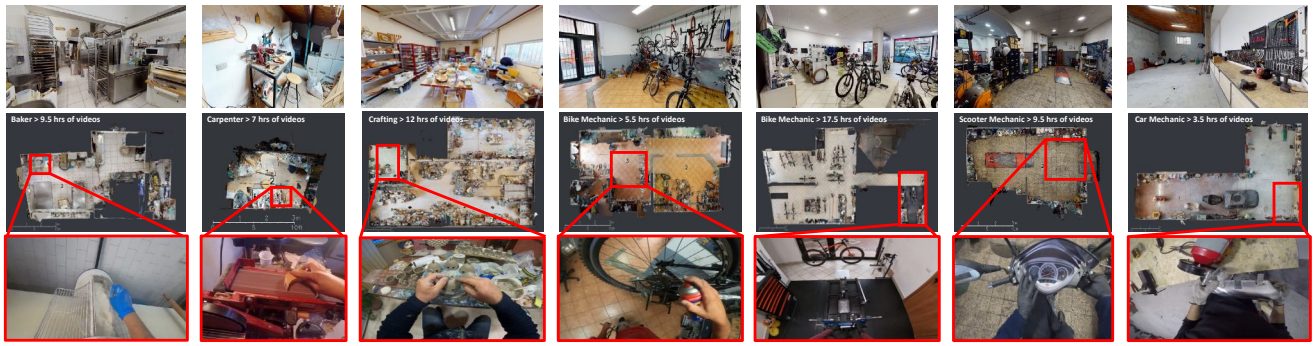


Figure 11. Matterport3D scans (top) related to seven different locations coupled with some videos (bottom).

setting. Thus, these data capture a single scene and social interactions from different points of view. We include both outdoor and indoor scenarios in Colombia. Outdoor scenarios include Bogotá and Cartagena's historical and colonial centers, as urban settings, and a Natural National Park and a stream, as rural settings. Indoor locations include professional activities such as laboratory workers and hair stylers. Furthermore, we include sports events such as salsa and urban dance rehearsals and rock climbing.

Primary contributors: Cristina González and Paola Ruiz Puentes.

Carnegie Mellon University, Pittsburgh, PA, USA and Kigali, Rwanda: Carnegie Mellon University (CMU) Pittsburgh gathered a large portion of its data from skilled workers such as carpenters, construction workers, landscapers, mechanics, arborists, painters, and artists. This portion of the dataset does not include any graduate students with the explicit goal of capturing a diverse range of real-world occupational activities. Over 500 hours of video were captured in the Pittsburgh area. The data was mostly recorded using a GoPro camera and a small portion was collected using WeeView, a wearable stereo camera.

Carnegie Mellon University Africa gathered data from hobbyist craftspeople and daily workers working in Kigali, Rwanda. An effort was made to collect data most representative of how tasks are carried out in Rwanda (such as doing laundry manually as opposed to with a washing machine). Over 150 hours of video were captured, and a portion of those hours are available in the current release. All of the data was collected using a GoPro camera.

Primary contributors: Kris Kitani - project coordinator for both CMU Pittsburgh and CMU Africa video collection. Sean Crane - lead coordinator of CMU Pittsburgh data collection (over 500 hours), main lead of CMU IRB review. Abrham Gebreselasie - lead coordinator of CMU Africa data collection. Qichen Fu and Xindi Wu - development of video de-identification pipeline, manual video de-identification annotation of CMU Pittsburgh data. Vivek Roy - main architecture of the license signing web server, coordinating with America Web Developers.

University of Catania, Italy: More than 359 hours of video have been recorded from 57 different subjects recruited through word of mouth, starting from family members, friends and acquaintances of students and faculty members of the research group. Videos are related to 25 scenarios. We chose the participants to cover a wide variety of professional backgrounds (24 backgrounds including carpenters, bakers, employees, housewives, artists, and students) and ages (subjects were aged from 20 to 77, with an average age of 36.42). 21 of the participants were female, while the remaining 36 were male. Female participants collected about 137 hours of video, whereas males collected 222 hours of video. The average number of hours of videos acquired by each participant is 6h:18m:23s, with a minimum number of hours of 06m:34s, and a maximum number of hours of 15h:40m:42s.

To prepare participants to record videos, we demonstrated to them the operations of the camera and how to wear it. We provided examples of valid recording and invalid recordings before they started the acquisition session. The recording procedure was described in a document left to the participants to help them remember the device usage and how to perform a good acquisition. Acquisition of videos has been performed using different models of GoPro cameras (GoPro 4, GoPro7, GoPro8, and GoPro Hero Max), which were handed over to the participants who typically acquired their videos autonomously over a period of a few days or weeks. 3D scans for 7 locations using the Matterport 3D scanner have been also collected (Figure 11).

Primary contributors: Giovanni Maria Farinella and Antonino Furnari - scenarios, procedures, data collection oversight, data formatting, encoding, metadata and ingestion. Irene D'Ambra - data collection, consent forms and information sheets, manual data review, de-identification oversight.

King Abdullah University of Science and Technology (KAUST), Saudi Arabia: A total of 453 hours of videos have been collected from 66 unique participants in 80 different scenarios with GoPro Hero 7. All the participants were KAUST community members, who are from various countries and have various occupations. All recordings took place in the KAUST university compound, which is 3600 hectares in area with diversified facilities (e.g., sports courts, supermarkets, a 9-hole golf course, and 2 beaches) and scenes (e.g., buildings, gardens, the red sea, and the desert). Therefore, the team was able to collect videos of various scenarios such as snorkeling, golfing, cycling, and driving.

The participants were recruited from multiple sources, such as friends and families, individuals referred to us by earlier participants, as well as people who were interested in our Facebook advertisements or posters in campus restaurants and supermarkets. Each candidate participant was required to register through an online form, which contained an introduction to and requirements of

the recording task, and collected his/her basic demographic information. The participants' ages range from 22 to 53. They come from 20 different countries, and about half are females. Many participants were graduate students and researchers, while others had various kinds of occupations such as chefs, facility managers, and teachers.

In order to prepare the participants for the recording process, the team described in documents and demonstrated to them the operations of the camera. The team also provided examples of what constitute valid and invalid recordings before they started. Each participant was provided a GoPro mountable camera with 2 batteries and a 512/256 GB SD card. Each participant needed to choose at least 2 different activities from our scenario list and record 1-10 hours of video within 2 days. The university team went through the recordings after the participants returned the camera to check their quality as well as to make sure the videos meet the university's IRB requirements.

Primary contributors: Chen Zhao, Merey Ramazanova, Mengmeng Xu, and Bernard Ghanem.

B. De-identification Process

The dataset has two types of video. The first includes videos recorded indoors where informed consent for capturing identities is explicitly collected from all participants in the scene, including faces and voice. Only video of this type is used in our Audio-Visual Diarization and Social Interaction benchmark studies. All 400 hours of data collected by Meta Reality Labs falls in that category. The second category, which forms the majority of our videos, requires de-identification as consent for capturing identities is not given—including footage captured outdoors in public spaces.[4] Only video collected by the universities falls into this second category. See Appendix A for details about the per-site collection approaches.

B.1 De-identification overview

All videos in the second category were manually screened to address any de-identification needs, and are further divided into two groups. Group1: videos that do not contain any personally identifiable information (PII).[5] This is when the video is recorded indoors with one person wearing the camera performing tasks such as cleaning or knitting for example, and no PII is present in the video. These videos did not require de-identification. Group2: videos where PII is captured. These include indoor settings with multiple participants present, PII captured accidentally such as an address on an envelope or a reflection of the wearer's face on a mirror or a surface, as well as videos recorded outdoors in a public space where bystanders or cars appear in the footage. Videos in Group2 were marked for de-identification, deploying advanced video redaction software, open source tools, and hours of human reviews to redact visible PIIs. University partners undertook this de-identification effort for their own data. We summarize the approach below.

Videos marked for redaction were processed through de-identification software that removes specific identifiers at scale. We used two commercial softwares: brighter.ai[6] and Primloc's Secure Redact[7] that enabled detecting faces and number plates automatically. We carefully reviewed all outputs from automated blurring, identifying both instances of false positives (blurring that mistakenly occurred on non-privacy related items) or false negatives (inaccurate or insufficient automated blurring of faces and number plates). Additionally, other PII data such as written names/addresses, phone screens/passwords or tattoos had to be manually identified and blurred per-frame. For this part of our de-identification process, we used both commercial tools within the above-mentioned commercial software and open source software, including Computer Vision Annotation Tool (CVAT)[8], Anonymal[9] and SiamMask[10].

[4]The exception is data from University of Minnesota, whose IRB permitted recording of incidental participants in public spaces having no manipulation or direct interaction with study personnel.

[5]We use the abbreviation PII to capture data protected under various data protection regimes including the General Data Protection Regulation (GDPR) where the term "personal data" is used.

[6]http://brighter.ai

[7]http://secureredact.co.uk

[8]https://github.com/openvinotoolkit/cvat

[9]https://github.com/ezelikman/anonymal

[10]https://github.com/foolwood/SiamMask

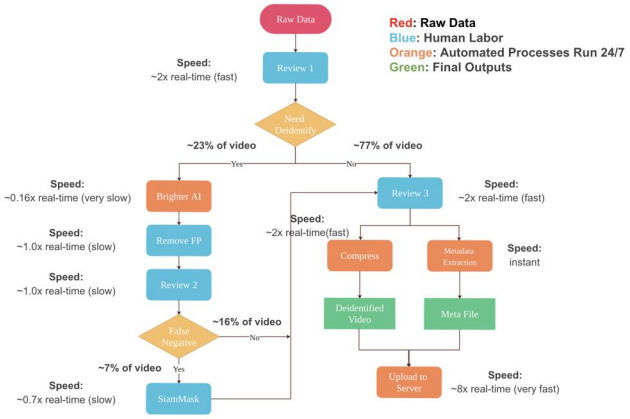


Figure 12. CMU's de-identification pipeline

Time costs. The relative time costs with respect to the original video length varied significantly for the different scenarios. Videos captured outdoors could take 10x the length of the video to carefully redact.

B.2 Sample pipeline

While partners followed varying pipelines, we offer a sample pipeline to showcase the process followed by Carnegie Mellon University that uses brighter.ai as the commercial software. This sample pipeline showcases the combination of automated processes and human labor with relative speeds of these steps.

This semi-automatic de-identification process was performed in four sequential stages (Figure 12): (1) automatic face and license plate detection, (2) false positive removal, (3) negative detection handling, and (4) image blurring.

Sensitive object detection Given the collected videos (raw data), a reviewer scans through videos and marks those containing sensitive objects such as human faces, license plates, credit cards, *etc.* Then de-identification software (brighter.ai) was used to automatically detect sensitive information.

False positive removal To improve the quality of the detection, false positives were removed. Reviewers manually scanned through the bounding boxes detected by the de-identification software, and rejected those bounding boxes which did not contain sensitive information.

False negative correction Additionally, reviewers studied every video to search for false negatives and manually annotated them using a bounding box. To make the process more efficient, an online object tracking algorithm [222] was used to generate bounding box proposals across frames. Reviewers verified that all tracked bounding boxes were correct.

Image blurring Once all of the detections were modified and corrected, a robust blurring process was used to de-identify image regions defined by the bounding boxes.

Time costs The relative time costs with respect to the original video length for each step are shown in Figure 12. Though this number depends greatly on the scenario captured in the video,

roughly speaking to de-identify 500 hours of video data, it took 780 hours of manual labor. Review 1 of 500 hours of video required 250 hours of work, removal of false positive over 115 hours of video took 115 hours of work, Review 2 of 115 videos took 115 hours of work, correcting false negatives in 35 hours of videos required 50 hours of work, and Review 3 of 500 hours of video took 250 hours of work (250+115+115+50+250 = 780 hrs).

C. Demographics

We further provide self-declared information on ethnic groups and/or country of birth by the participants. We report these separately per state/country due to the differences in granularity of ethnic groupings. All participants are residents in the country specified per paragraph. This data is not available for participants from Minnesota, US.

United Kingdom Residents Reporting demographics was optional and thus 63% of participants (52/82) that reside in the United Kingdom self-reported their ethnic group membership as follows:

White — English, Welsh, Scottish, Northern Irish or British	35
White — Any other White background	12
Mixed — White and Asian	1
Mixed — Any other Mixed or Multiple ethnic background	2
Arab	1
Prefer not to say	1

Italy Residents 100% of participants that reside in Italy self-reported their country of birth as follows:

Italy	53
Germany	1
Russia	1
Portugal	1
Poland	1

India Residents 100% of participants that reside in India self-reported their ethnic group membership as follows:

Eastern India	10
Northern India	15
Southern India	108
Western India	5

Pennsylvania, USA, Residents 100% of participants that reside in Pennsylvania, USA, self-reported their ethnic group membership as follows:

White	42
Asian	4
Mixed — White and Black African	2
Black, African, Caribbean	1

Washington, US, Residents 100% of participants that reside in Washington, USA, self-reported their ethnic group membership as follows:

Caucasian	101
Black or African American	58
American Indian (Native American)	24
Hispanic	19
Indian (South Asian)	4

Indiana, US, Residents 95% of participants that reside in Indiana, US, self-reported their country of birth as follows:

US	39
China	10
India	10
Bangladesh	2
Vietnam	2

Georgia, USA, Residents 100% of participants that reside in Georgia, USA, self-reported their ethnic group membership as follows:

White / Caucasian	16
Black / African American	1
Asian / Indian & White / Caucasian	1
Other / Taiwanese	1

Japan Residents 100% of participants that reside in Japan self-reported their ethnic group membership as follows:

Asian (Japanese)	81

Kingdom of Saudi Arabia Residents 100% of participants that reside in KSA self-reported their country of birth as follows:

China	12
Russia	9
Colombia	8
Mexico	5
Kazakhstan	4
India	4
US	4
Saudi Arabia	3
Kyrgyzstan	2
New Zealand	2
Greece	2
Ukraine	2
Italy	2
Lebanon	1
Jordan	1
Egypt	1
Kashmir	1
Portugal	1
South African	1
Thailand	1

Singapore Residents 100% of participants that reside in Singapore self-reported their nationalities as follows:

Chinese	26
Singaporean	12
Indian	1
Malayan	1

Colombia Residents 90% of participants that reside in Colombia self-reported their ethnic group membership as follows:

Hispanic/Latin	62
White/Caucasian	4
Black, African or Caribbean	1
Mixed - White an African	1
Prefer not to say	1

Rwanda Residents 100% of participants that reside in Rwanda self-reported their ethnic group membership as follows:

Black, African or Caribbean	14

D. Narrations

The goal of the narrations is to obtain a dense temporally-aligned textual description of what happens in the video, particularly in terms of the activities and object interactions by the camera wearer. The Ego4D narration data is itself a new resource for learning about language grounded in visual perception. In addition, as described in the main paper, we leverage the narrations as a form of "pre-annotation" to index the videos by semantic terms. Specifically, the narrations are used to construct action and object taxonomies to support various benchmarks, to identify videos that are relevant to each benchmark, and to select regions within the videos that require annotation.

This section overviews how we instructed annotators to narrate the videos, and how we transformed narration text into taxonomies of objects and actions.

D.1 Narration instructions and content

We divide the dataset into clips of (max) 5 minutes long when acquiring narrations. Each 5-minute clip is then passed to two different annotators, to collect two independent sets of narrations for every video clip in the dataset for better coverage and to account for narration errors.[11] Narrators are instructed to watch the 5 minute video clip first, and then asked to provide a short 1-3 sentence "summary" narration for the entire clip that corresponds to the overall activity and setting of the video clip (e.g., "the person does laundry in the washing machine"). These summaries are marked with the tag "#summary" in the released narrations.

Following this first screening, which is critical for the overall understanding of the clip, the dense narrations are collected as follows. Annotators re-watch the clip, pause and mark the timepoint when something happens in the video, then enter a short natural language description of the ongoing action or interaction, before resuming watching the video.

Narrators are provided the following prompt: *"Pretend as you watch this video that you are also talking to a friend on the phone, and you need to describe to your friend everything that is happening in the video. Your friend cannot see the video."* This prompt is intended to elicit detailed descriptions that provide a play-by-play of the action. See Figure 13 for an illustration of the narration tool interface. Each narration thus corresponds to a single, atomic action or object interaction that the camera wearer performs (e.g., "#C opens the washing-machine" or "#C picks up the detergent", where the tag #C denotes the camera wearer). Importantly, our narrations also capture interactions between the camera-wearer and others in the scene, denoted by other letter tags, e.g. #X (e.g. "#C checks mobile while #X drives the car", "#C passes a card to #Y"). See Figure 14 for narration examples.

D.2 Narration analysis

We present some statistics on the collected narrations. Altogether, we collected 3.85M sentences across the 3,670 hours of video. Figure 15 (left) shows the distribution of frequency of narrations across all videos in the dataset. Depending on the activities depicted, videos are annotated at varying frequencies. For example, a video of a person watching television is sparsely annotated as very few activities occur (0.17 sentences/minute), while a video of a person harvesting crops, performing repetitive actions is densely annotated (63.6 sentences/minute). On average, there are an 13.2 sentences per minute of video.

Figure 15 (middle and right) show the distribution of length of the collected narrations. The individual timepoint narrations are short, highlight a single action or object interaction, and have an average of 7.4 words. Though short, these narrations cover a variety of activities ranging from object interactions, tool use, camera wearer motions, activities of other people etc. In contrast, the summary narrations are longer (on average, 16.8 words) and describe activities at a higher level. Table 2 shows a few text examples of each type of narration in addition to the visual examples in Figure 14.

Finally, we study the diversity of the video dataset by looking at the frequency of occurrence of words in the narrations collected for videos of each scenario type. Figure 16 shows word clouds depicting objects that prominently feature in across various scenarios. The word clouds highlight characteristic objects per scenario (e.g., bowl, spoon, plate in "Cooking" videos; card, dice, pawn in "Playing board games" videos) while also hinting at common objects across all scenarios (e.g., hands, paper, phones). The diversity in narrations collected highlights the diversity of video content captured in the dataset.

D.3 Action and object taxonomy

In total the raw narrations describe the Ego4D video using 1,772 unique verbs and 4,336 unique nouns. The distribution of the most frequently occurring verbs and nouns can be seen in Figure 17.

Following ideas from [44], we leverage the narrations data to construct a taxonomy over the actions and objects that appear in the video, as follows. We use a part-of-speech (POS) tagger and dependency parser to identify verbs and nouns from each narrated action. We use an ensemble of parser models from the Spacy [98] toolkit to do this. Given a natural language narration, we first identify verbs using their POS tag. Then using the dependency tree, we identify all direct objects of the verb. To ensure verbs and nouns are accurately parsed, we adopt several heuristics: Parsed verbs are split into multiple senses (e.g., "turn" is split into "turn-on", "turn-off" and "turn-over"); compound nouns are decomposed into a root noun coupled with a modifier to ensure the noun taxonomy is unambiguous (e.g., modifier "egg" and root noun "shell" in "egg shell"); collective nouns are mapped to their main entity (e.g,. "piece of cheese" $\rightarrow$ "cheese"). Finally, we manually cluster the verbs and nouns to avoid redundancy in the taxonomy (e.g., "cut", "chop", "slice" are all mapped to the verb cluster "cut").

The resulting taxonomy consists of a set of 115 verbs ($\mathcal{V}$) and a set of 478 nouns ($\mathcal{N}$). Figure 39 shows the distribution of verbs and nouns in a set of video data annotated with the taxonomy. See Section J.2 for details on how the taxonomy is used in the context of the benchmark tasks.

[11] We simply keep both independent narrations; they are not merged because they do not serve as ground truth for any benchmark.

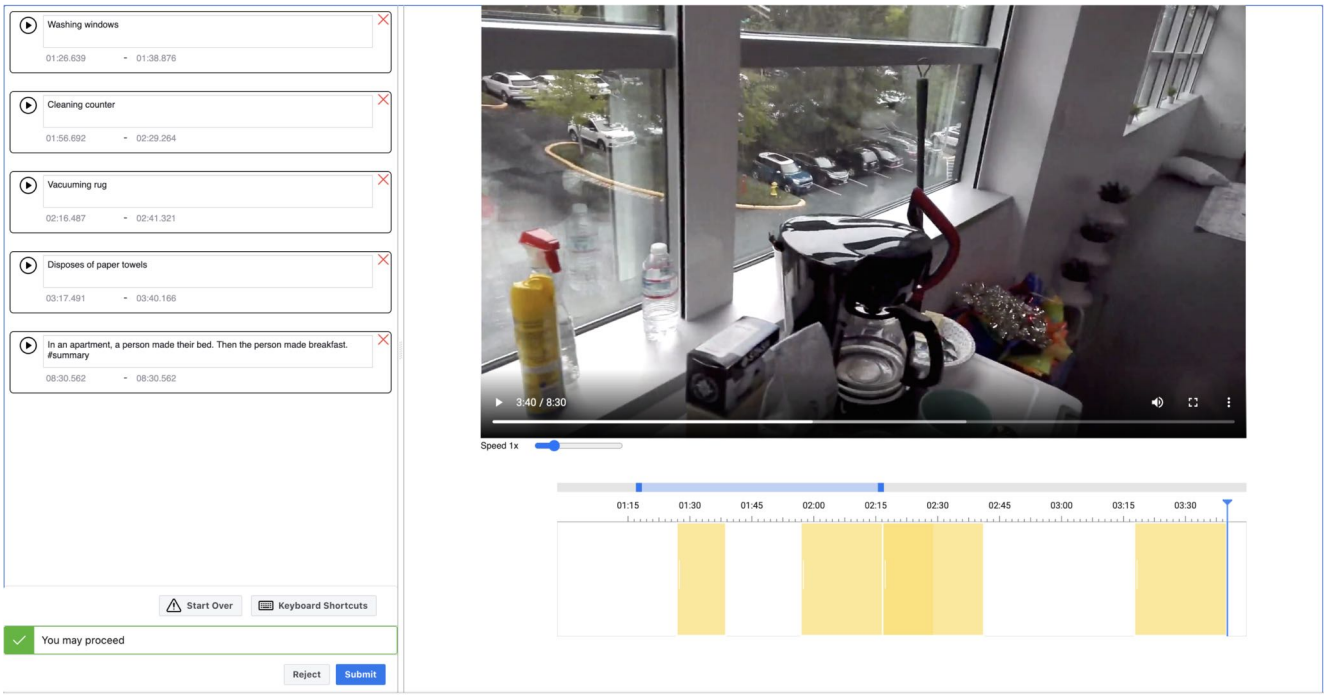


Figure 13. **Narration tool interface.** Narrators mark a timepoint where something happens in the video (bottom bar), and enter a text description of the activity (left sidebar).

Object interaction	**Context objects**	**Multi-person actions**	**Manipulation actions**
#c c flips the paper	#c c taps a hand on the floor	#o a man x moves the legs.	#c c cuts a leaf from the plant with his left hand.
#c c lifts the t-shirt	#c c holds the wheel with his left hand.	#o a man y sits on a chair	#c c pulls his hand off the chess piece
#c c drops the plate	#c c puts the brush in the colours.	#o a woman x steps forward.	#c c holds the knitting needle with the other hand
#c c holds the piece of cloth	#c c places plastic models kit on the table	#o a person x hits the cricket ball	#c c opens the screwdriver container with his hands
#c c fixes on the model craft	#c c arranges the doughs on the tray	#o a man y throws the ball towards man x	#c c touches the piece of wood with the hand
Camera wearer motion	**Summary narrations**		
#c c raises hands	c was in a room,fixed a wood model kit. #summary		
#c c stands	c tightened the motor on the head of the hoe of the lawn mower. c cut grasses on the field with the lawn mower. #summary		
#c c stands up from the stairs	c was in a kitchen, he cut sausages in to pieces with a knife, mixed the sausages and cooked them with a pan. #summary		
#c c walks around a kitchen	c was in the house and she studied #summary		
#c c sits up	c studied in a room. c went through a mobile phone and a mobile tablet while reading in the room. #summary		

Table 2. **Text examples of narrations.** The collected narrations describe diverse aspects of human activity. Summary narrations capture high level descriptions of activities in a 5 minute clip. See Figure 14 for visual examples.

D.4 Narrations for annotation prioritization

All videos in Ego4D are narrated, and subsets of them are manually labeled for each benchmark. Rather than randomly label instances for a given benchmark, we aim to target those that are most relevant to the task. For example, videos likely to contain multi-person conversation are most interesting for the AV Diarization benchmark, whereas videos with ample hand-object interaction are most interesting for Hands and Objects. To that end, we use the narrations and summaries as a tool to automatically prioritize certain videos to label per benchmark. The benchmark appendices below provide details.

D.5 Contributions statement

Tushar Nagarajan developed the taxonomy, helped develop narration instructions, and performed the narration analysis presented in the paper. Kristen Grauman developed narration instructions, helped coordinate pilots and annotation work, and contributed to taxonomy formation. Michael Wray co-developed the taxonomy.

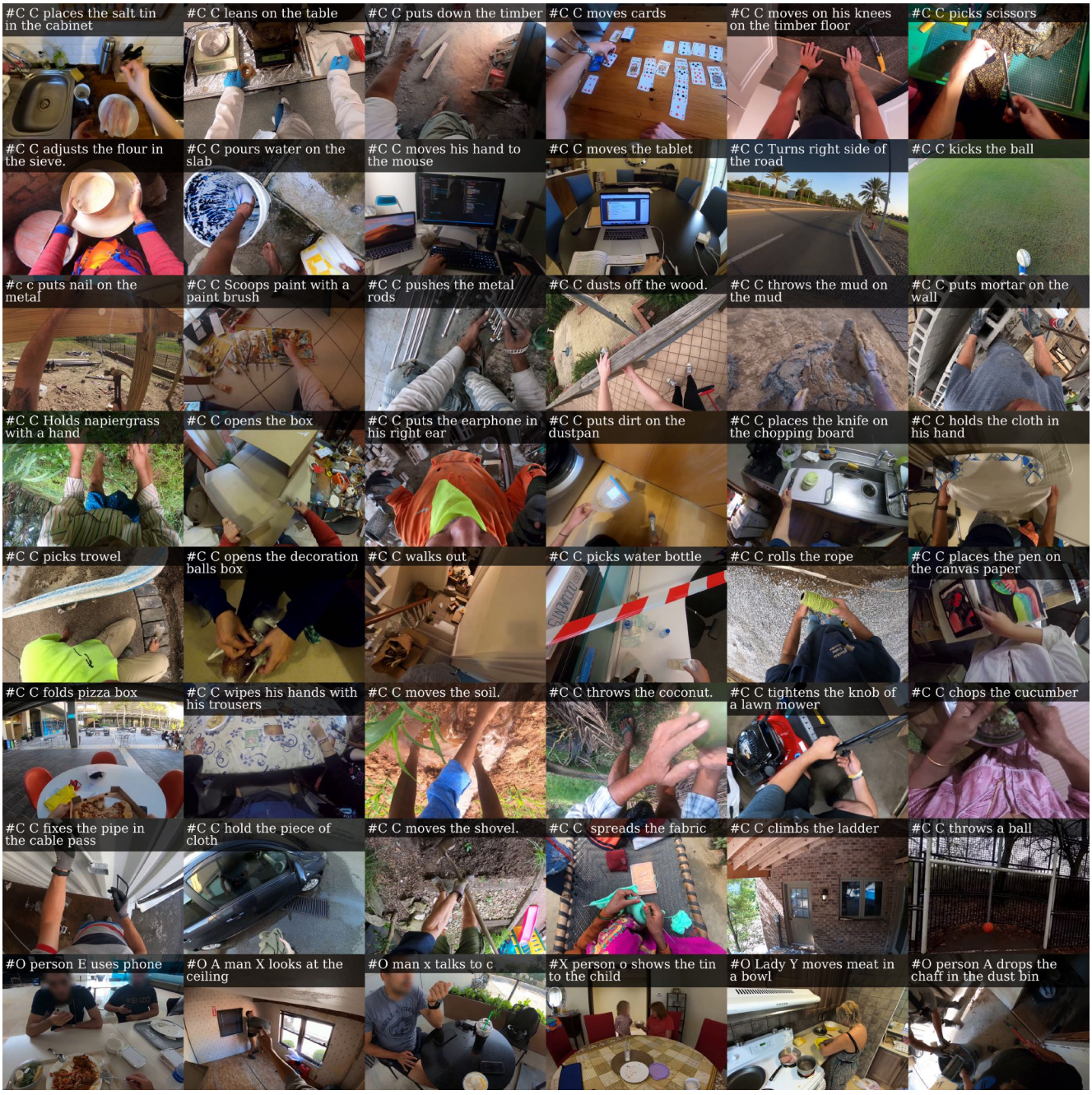


Figure 14. **Example narrations at keyframes of video.** #C refers to the camera-wearer. The last row shows narrations that include other people that participate in activities with the camera-wearer (denoted by other letter tags, e.g., #O, #X).

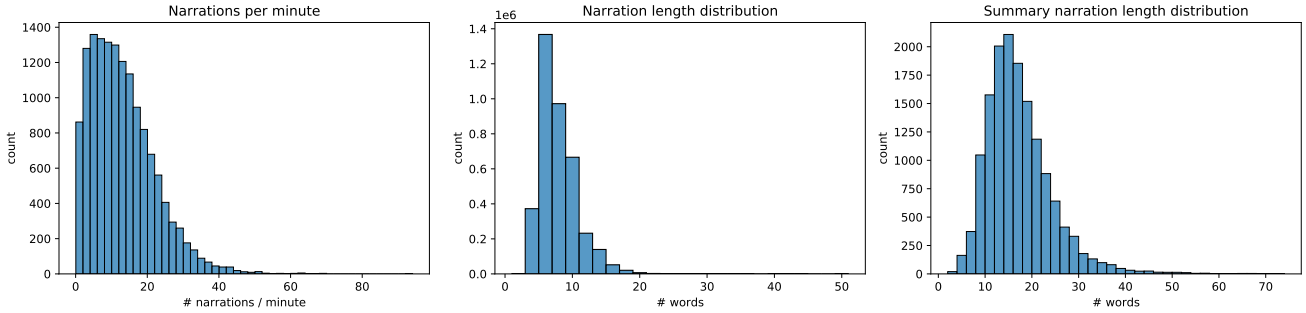


Figure 15. **Collected narration statistics. Left:** Distribution of frequency of narrations collected. **Middle and right:** The distribution of length of the collected narrations and summaries. Summaries are naturally longer, and describe activities at a higher level compared to individual action narrations. See text for discussion.

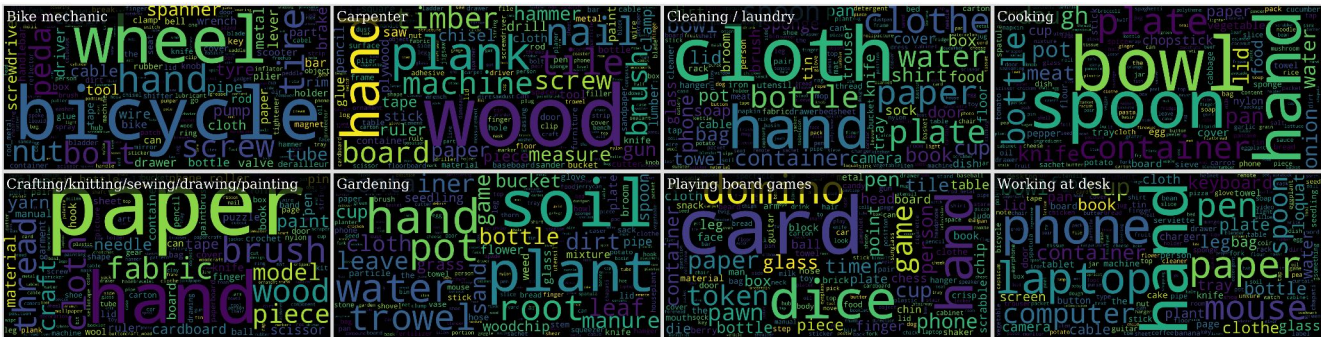


Figure 16. **Distribution of objects in narrations of videos from eight common scenarios.** The variety of objects covered across scenarios showcases the diversity of activities in the video collected.

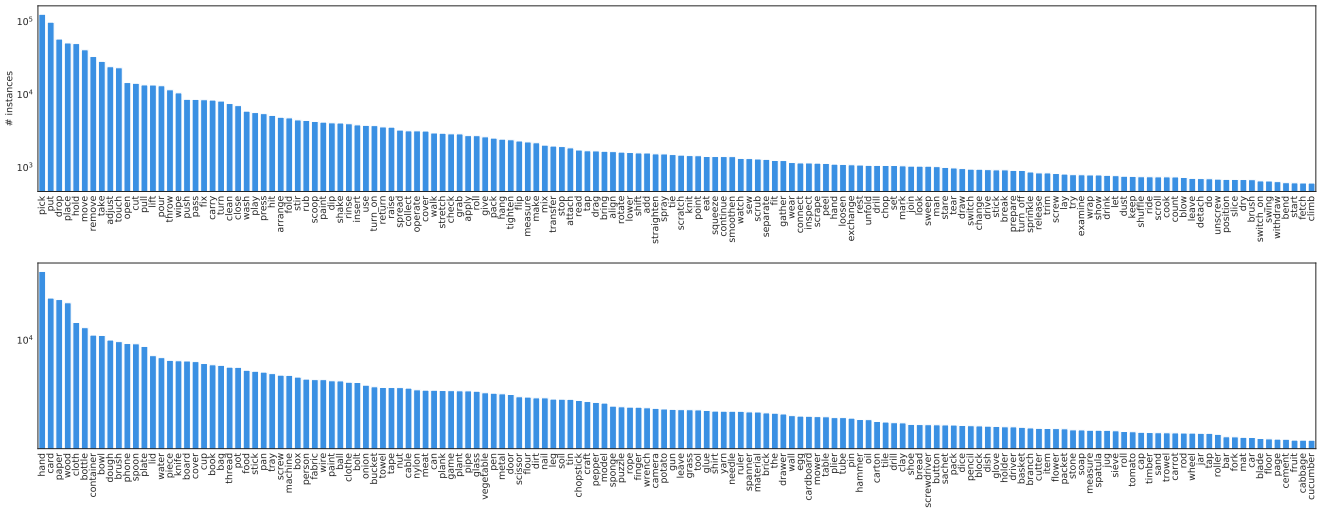


Figure 17. **Narration verb/noun distribution.** Distribution of automatically extracted verbs (top) and nouns (bottom) from narrations. Top 150 most frequently occurring of each is shown for clarity.

E. Benchmark Data Splits and Accessibility

For each benchmark task, certain portions of the Ego4D video repository are labeled. Table 3 shows the breakdown of the amount of data annotated for each. Note that there are 764 total hours of video relevant to the AVD and Social tasks (i.e., have audio, conversation, and unblurred faces), including the annotated set of 47.7 hours above. For other benchmarks, the relevance has a softer dependency on the specific video content (e.g., a memory query can apply to any of the 3,670 hours). The following appendices will explain how we sampled data to be annotated for each benchmark.

For the public Ego4D benchmark challenge, we ensure that the splits are consistent within a family of related tasks. For instance, all the Forecasting and Hands+Objects tasks share the same splits and ensure training videos in one do not occur as validation videos in another. Similarly, the Episodic Memory tasks share the same splits. However, it is harder to ensure this across very different tasks, since the videos selected for annotations are different. For example, the Social benchmark considers multi-person interactions which may not have many hand-object interactions; hence the set of videos labeled for Social and Hands+Objects have little overlap and the train/val/test splits are naturally different.

Since we plan to use the test set for the public challenge, we are withholding all the test annotations and making them accessible only through a submission server. We are also withholding the narrations that overlap with any of the test sets.

At 3,670 hours of video, we are mindful that Ego4D's scale can be an obstacle for accessibility for some researchers, depending on their storage and compute resources. To mitigate this, we have taken several measures. First, we provide precomputed action features (SlowFast 8x8 with ResNet 101 backbone pretrained for Kinetics 400) with the dataset, an optional starting point for any downstream work. Second, only portions of the data constitute the formal challenge train/test sets for each benchmark—not all 3,670 hours (see Appendix E). As Ego4D annotations increase, we will create standardized mini-sets. Finally, we provide the option to download only the data targeting an individual benchmark or modality of interest.

	Num hours	Num clips	Avg clip length
EM VQ-2D	432.9	5,831	6.1 min
EM VQ-3D	13	159	4.9 min
EM Moments	328.7	2,522	7.9 min
EM NLQ	227.1	1,659	8.2 min
Hands+Obj.	196.2	88,585	8.0 sec
Forecasting	110.5	1,498	4.4 min
AVD	47.7	572	5 min
Social	47.7	572	5 min

Table 3. Amount of annotated data for each benchmark. EM refers to Episodic Memory and AVD refers to Audio-Visual Diarization. All 3,670 hours of video have narrations and features.

F. Episodic Memory Benchmark

This section details the Episodic Memory benchmark task definitions, annotations, baseline models, and results.

F.1 Formal task definitions

As presented in the main paper, there are three kinds of Episodic Memory queries—visual, natural language, and moments—each of which requires localizing the response in the video. Their formal definitions are as follows.

Visual queries (VQ) This task aims to query an egocentric video based on a static image crop of an object. Specifically, it asks the question 'Where was object X last seen in the video?', where X is a single 'canonical' image crop in which the object is clearly visible and human-identifiable. A potential use case for visual queries is where a user teaches the system a new object by showing a photo ("these are my keys") and then later queries for it among past video. By enabling visual queries, as opposed to categorical queries, this is a form of open-world object localization.

We formulate the problem as follows. Given an egocentric video $\mathcal{V}$, a query object o specified via a static visual crop v, and a query frame q, the goal is to identify when the object o was last seen in the video before the query frame q. The response is specified as a 'response track' r which is a temporally contiguous set of bounding boxes surrounding the object o in each frame:

$$r = \{r_s, r_{s+1}, \cdots, r_{e-1}, r_e\}, \tag{1}$$

where s is the frame where the object o (at least partially) enters the camera-wearer's field of view, e is the frame where the object exits the camera-wearer's field of view, and r_i is a bounding box (x, y, w, h) in frame i. If the object appears multiple times in the video, the response only refers to the 'most recent occurrence' of the object in the past, i.e., the response track which minimizes $q - r_e$ with $q > r_e$.

When a 3D scan of the environment associated with the video is available, the response additionally includes a 3D displacement vector $\Delta d = (\Delta x, \Delta y, \Delta z)$ between the 3D location where the query was made (i.e., at query frame q), and the 3D location in the environment where the object was last seen (i.e., at the end of the response track r_e).

Natural language queries (NLQ) The motivation behind the NLQ task is to enable searching through an egocentric video using a natural language query. The system responds to a query by providing a temporal window localized in the video, from which the answer to the query can be deduced. These queries can be related to objects, places, people, and activities that appeared in the episodic memory of the user. Note that we only consider episodic queries, *i.e.*, queries that can be answered/deduced from the egocentric videos, and not factual queries, *i.e.*, queries that require an external knowledge base to answer.

NLQ is a challenging multimodal task requiring visual and linguistic understanding and reasoning. Consider the query *"What did I pick up before leaving the party?"* In order to fulfill this request, the system needs to: (a) break down and understand the language query as a search for an object (*what*) with which the user interacted (*pick up*) before an event (*leaving the party*), (b) go through the egocentric video and identify the desired event of *"leaving the party"*, (c) visually search for the object with which the user interacted prior to this event. This example demonstrates the complexity of NLQ from both visual (recognizing events, objects, places, *etc.*) and linguistic (breaking down reasoning, understanding relations, *etc.*) perspective. In addition, the diverse set of queries within NLQ, while facilitating a flexible search and retrieval through an intuitive interface of language, also increases the complexity of the task.

Concretely, NLQ is formulated as follows: Given an egocentric video $\mathcal{V}$ and a natural language query $\mathcal{Q}$, the goal is again to identify a 'response track' r, such that the answer to $\mathcal{Q}$ can be deduced from r. The response track should be a set of temporally contiguous frames within $\mathcal{V}$. Given the episodic nature of our task, r should be sufficient to answer $\mathcal{Q}$, without the additional need for $\mathcal{V}$ or any external knowledge bases.

Moments queries (MQ) This task aims to query an egocentric video based on a category of actions. Specifically, it poses the following request 'Retrieve all the moments that I do X in the video.', where 'X' comes from a pre-defined taxonomy of action categories, such as 'interact with someone' or 'use phone'. Compared to the natural language queries, the moment queries focus on daily-life actions or activities. One moment query can correspond to multiple response instances (temporal windows) in the video. This task provides the user a fast and convenient way to retrieve multiple action moments at a time, where the user does not need to come up with a sentence to describe what he/she wants, but instead can directly choose among the pre-defined categories.

The moment queries task is related to the task of temporal action detection [141, 229, 237], which aims to identify and localize all instances of all action categories that take place in a video. Both tasks have a list of action categories pre-defined, and both aim to predict multiple action instances with their temporal boundaries. The difference is that 1) our moment queries task is a retrieval task where action categories are provided as queries, meaning it does not need to produce instances of categories that are not among the queries; and 2) our moments taxonomy is specific to first-person activity. We aim for moments that are activities at a medium level of granularity—coarser than the actions in Forecasting, and finer than the "scenario" labels shown in Figure 3 of the main paper.

The MQ task is also related to temporal language grounding in videos [236], which aims to retrieve a segment from a video, as queried by a natural language sentence. Both tasks have a query and aim to predict corresponding temporal segments. The difference is that MQ uses pre-defined query categories rather than natural language sentences, and one query can correspond to multiple instances rather than a unique one.

We formulate the problem as follows. Given an egocentric video $\mathcal{V}$, and a query action category c, the goal is to retrieve all the instances of this action category in the video, assuming that the query is made at the end of the video. The response is a set of action instances of the category c $\Phi_c = \{\phi_n = (t_{n,s}, t_{n,e}, s_n)\}_{n=1}^{N}$, where n is the number of instances for this category, $t_{n,s}$ and $t_{n,e}$ are start time and end time of the n^{th} instance respectively, and s_n is its prediction confidence.

F.2 Selecting clips for annotation

For all benchmarks we sample video clips to annotate based on criteria for geographic diversity and scenario diversity. For Episodic

Navigation verbs for entropy-based video selection					
appear	ascend	bend	bring	carry	catch
climb	close	come	descend	dig	dispose
drag	dribble	drop	enter	fall	fetch
find	fly	gather	get	give	grab
hang	jog	jump	kick	lean	leave
lift	lower	move	navigate	open	propel
raise	return	ride	rise	run	shut
steer	step	turn	vaccum	walk	

Table 4. We prioritize videos to annotate for visual queries based on the entropy of these navigation-related verbs in the narrations.

Memory we impose additional sampling criteria meant to highlight data most interesting for the task, as follows.

Visual queries Video clips to annotate for visual queries (VQ) are selected based on the frequency of object occurrences and amount of navigation in the video. To have interesting visual queries in a video, there must be several 'interesting' objects that can be queried about. An object is 'interesting' in the context of visual queries if there is a sufficiently high separation in space and time between any two occurrences of the object. This typically happens when the camera-wearer visits the location near the object briefly, and then navigates elsewhere before revisiting the object again. For example, consider a person who finishes cleaning a living room, visits the kitchen for some period of time before revisiting the living room again. Most objects in the living room are interesting to query about when the person is in the kitchen.

To select videos based on these considerations, we use a two-step process. First, we filter out videos based on the associated 'scenario' labels (see Figure 3) that provide high-level information about the content and activities in videos (e.g., cooking, cleaning, golfing, etc.). We manually preview randomly sampled videos from each scenario to identify interesting scenarios such as cooking, indoor navigation, farmer, cleaning, and grocery shopping. We then sort videos within each scenario based on a scoring function using the narrations for the video. Specifically, we extract the list of verbs in the narrations (along with their frequencies). We then measure the entropy of the distribution of manually curated *navigation* verbs (See Tab. 4). The video is more likely to allow challenging visual queries if its navigation entropy is higher. For videos with near-zero entropy, we observe that the camera-wearer is usually staying static in a single location without any movement. Finally, a limited number of 3D scans were available for the 3D localization task. Videos associated with these scans were prioritized, regardless of their navigation entropy, in support of the 3D response version of the VQ task.

Natural language queries For NLQ we apply similar sampling criteria as above for VQ, but augment it to avoid repetitive actions (e.g., sewing while sitting on the couch). First, we manually select amenable scenarios (see Figure 3). Among those, we prioritize clips with high entropy computed over navigational terms as above. Finally, we prioritize non-repetitive actions by computing the ratio of the number of unique verbs in a clip's narration vs. the total number of verbs in that same narration—higher is better.

Moments queries To select clips for moments queries, we compute the overlap of verbs/nouns with the moments taxonomy. We calculate a similar entropy-based score and sort videos according to this score. In addition, we restrict videos to a fixed set of categories present in our taxonomy to avoid labeling videos that do not contain relevant activities.

F.3 Annotation

Next we describe the annotation procedures and outputs for Episodic Memory.

Visual queries For annotating visual queries, we first sample contiguous clips of varying lengths (5 mins, 8 mins, and 16 mins) from the set of interesting videos. The annotators are instructed to create and annotate 3 visual queries for each clip. A visual query consists of the query frame q, the visual crop v of the query object o, the response track $r = \{r_s, r_{s+1}, \cdots, r_{e-1}, r_e\}$, and a textual name for the object (eg. cup, hammer, broomstick, etc). The annotators performed the following steps to annotate a given clip:

1. Identify three interesting query objects in the clip. An object is interesting if it occurs in at least two different parts of the video.
2. For a given object, enter a textual name. While our current task queries with the image crop, not the name, this annotation will allow future variants that do query for the object by name.
3. Select one of the object occurrences in the video and mark a visual crop $v = (x_v, y_v, w_v, h_v)$. The visual crop must be a good representative view of the object, and it must have good lighting, large-enough size, and must not be blurred.
4. Mark a *different occurrence* of the object as the response track $r = \{r_s, \cdots, r_e\}$. The response track starts from the frame when the object is first visible and ends when the object leaves the field-of-view. The response track must also be contiguous in time and the bounding boxes must accurately mark the position and size of the object.
5. The query frame q is sampled some time *after* the response track r. The object o must not appear anywhere between the response track r and the query frame q, so that the ground truth is well-defined and unique for "when did I last see...?".

For each annotation, we apply automated and manual quality checks to ensure correctness. In case the quality falls below a certain threshold, the clip is reannotated.

For visual queries associated with 3D scans, we also collect 3D annotations in the form of 3D bounding boxes capturing where the object was last seen. We then use those bounding boxes to establish the ground truth displacement vector from the query frame to the object, which is the target of the task. Each annotation a_q is collected in the scan coordinate system s:

$$T_s = [R_s | t_s], \tag{2}$$

where $q \in \{1, \ldots, \mathcal{Q}\}$, $\mathcal{Q}$ the total number of queries, and where $T_s \in \mathbb{R}^4$ is the transformation matrix of the bounding box. R_s and t_s are the corresponding rotation and translation for annotation a_q.

The annotation procedure is defined as follows: A query consists of a video clip, a visual crop, and a response track. For each query,

the goal is to retrieve in the scan the location of the object defined in the video. Once the location is found, we draw a 3D bounding box at this position with the appropriate scale and orientation. It is important to note that 3D scans and videos have been recorded at different times. Therefore, it is likely that an object at a certain location in the video will not be present at that same location in the 3D scan. In such cases, we ask the annotator to hallucinate a 3D bounding box in the 3D scan at the position of the target object defined in the video.

In order to validate an annotation we collect two 3D bounding boxes per query from two different annotators. Leveraging the two boxes we compute the following validation metrics:

$$d_{norm} = \frac{\|c_1 - c_2\|_2}{m_{diag}} \tag{3}$$

$$V_{norm} = \frac{V_{global}}{V_{union}}, \tag{4}$$

where c_1 and c_2 are the centroids of the two boxes, m_{diag} is the average diagonal length of the two boxes, V_{global} is the volume of the 3D convex hull of the two boxes, and V_{union} is the volume of the union of the two boxes. These metrics measure the agreement level betwen the two annotators. When the two annotations are perfectly aligned, the metrics are equal to $d_{norm} = 0$ and $V_{norm} = 1.0$. The assumption is that if the two annotators agree on the position, scale, and orientation of the bounding box then it is likely to be correct. If the two annotations are far from each other we will discard the query. There are a couple of reasons that can explain such case: (1) one annotator mislabeled the query, (2) the query is hard to annotate. Some queries require a significant amount of hallucination to retrieve the object location in the scan which clearly leads to subjective annotations. We empirically defined two thresholds of 1.5 over d_{norm} and 15 over V_{norm} to filter out poor annotations. Any query that has either one of the two metrics above the threshold of acceptance is rejected.

Natural language queries To collect NLQ annotations, we sample contiguous clips of length 8 minutes and 20 minutes. The annotators are instructed to watch these clips and generate natural language queries, focused on retrieving information about objects, places, and people in the egocentric video clips. To reduce the cognitive overload on the annotators, and focus their efforts on memory-relevant queries, we also provide a list of 13 query templates (see Table 5), corresponding to queries a user might ask to augment their memory. Note that these templates are provided only to guide their choice of query, and does not limit the linguistic variability since the annotators are instructed to paraphrase the template without copying them as is.

To elaborate, the annotators performed the following steps:

1. Watch the entire video clip $\mathcal{V}$ in order to understand the high-level context (optionally in $2\times$ fast-forward),
2. Pick a query template from the available list and paraphrase/reword the query to obtain $\mathcal{Q}$, *e.g.*, template *'Where was object X before/after event Y?'* can be paraphrased as *'Where was the blue bucket prior to my dog exiting the living room?'*
3. Find the temporal window where the response to the natural language query can be deduced visually, and annotate it as r.

Category	Template
Objects	Where is object X before / after event Y?
	Where is object X?
	What did I put in X?
	How many X's? (quantity question)
	What X did I Y?
	In what location did I see object X ?
	What X is Y?
	State of an object
	Where is my object X?
Place	Where did I put X?
People	Who did I interact with when I did activity X?
	Who did I talk to in location X?
	When did I interact with person with role X?

Table 5. The NLQ templates capture a diverse set of queries that humans can ask to augment their memory and recollect objects, places, and people in their everyday experience.

During our data collection, we also requested the annotators to mark the slot values and corresponding verbs, for the selected language query templates. While we do not use this information for our task, it may be useful for other future research.

The desiderata for the collected queries are as follows. They should: (a) reflect the underlying motivation of augmenting human memory, (b) be rich and diverse in terms of language and the objects, places, people, and events, and, (c) be challenging enough for an intelligent system but not too complicated or convoluted to reduce the naturalness of the queries. For instance, though a query like *'What was playing on the television when I was folding my seventh T-shirt after my dog exited the room?'* is challenging from a learning perspective, it is not natural from an application standpoint. In order to ensure the above qualities for NLQ, we enforce the following constraints:

- All paraphrased language queries must be in past tense, and must be posed as questions asked at the end of the entire video clip. This resembles the real-life scenario of querying about episodic memory (past) of the user, and resolves ambiguity when there are multiple occurrences of an object to the the last relevant one.
- To account for momentary shifts of view for the egocentric video, we allow small interruptions (< 3 seconds) between the truly relevant frames for a given query. In other words, frames where the object/person/place of interest goes out of view for less than 3 seconds as a result of momentary gaze shift are still considered to be contiguous.
- For a given query, if there are multiple non-contiguous temporal windows (separated by more than 3 seconds) as independently valid answers, we instruct the annotators to either discard the query and create a different one, or add more details to the wording to make it more specific. Similarly, queries that require multiple temporal windows (separated by more than 3 seconds) to deduce the answer are also disallowed. For example, *'How many shirts did I pack in my*

suitcase?" is invalid if packing happens across multiple temporal windows, separated by more than 3 seconds (*e.g.*, the user pauses to make coffee, and then returns to packing).

- We encourage diversity by instructing that the query responses not be concentrated at one part of the video clip, or around few objects/places/people. In addition, we also disallow the query response window to be more than 50% of the total clip length.

- Finally, queries that require reasoning and knowledge on top of visual evidence are invalid. For instance, *'What country's flag was hanging on the wall?"* is invalid while *'Where was the flag that was hanging on the wall?"* is valid. Similarly, queries that guess the motivation or intentions of the user or people in the video clip are also not allowed. As an example, *'Why did the person at the door leave a package on the porch?'* is disallowed while *'What did the person leave on the porch?'* is accepted.

After the annotation process, we apply both automatic and manual quality checks, including the diversity of language queries and temporal window locations, to score the annotations. If the overall quality score is below a threshold, the clip is re-annotated.

Moments queries To annotate moments queries, we sample contiguous clips of 8 minutes from the set of interesting moments videos. The annotators are instructed to mark instances of activities with a temporal window and the activity's name from a fixed taxonomy of activities. We have each instance labeled by three independent annotators. By assuming each annotator is reliable, we take the union of moments across annotators to ensure completeness of annotations.

The taxonomy was created semi-automatically from the narrations. Specifically, we use the *summary* narrations collected for five-minute clip segments, as they capture higher-level events and activities that are suitable for the moments retrieval task. This is in contrast to the verb-noun taxonomy that is sourced from individual narrations for each atomic action, which are used in the Forecasting and Hands and Objects benchmarks (see Appendices G and J).

The taxonomy was created as follows. First, each summary narration was encoded into a feature vector using a pre-trained BERT [51] language model, and then concatenated with the word embeddings for the main verb and noun extracted from the summary. These summaries were then clustered into groups, and then labels were manually assigned to groups based on the coherent activities they described.

Note that this process was done independently for a set of scenarios that we selected based on how frequently they occur in the dataset, the diversity of activities they represent, and how likely they contain high-level, event-like activities. For example videos that primary involve a single activity like "driving" are not interesting categories in this context, whereas "household cleaning" contains several different activities that are shared across other indoor tasks, making it an appropriate scenario. In total, we select videos from 5 scenarios to create our moments taxonomy: Cooking, Cleaning, Shopping, Handyman, Farmer/Gardener. Each annotation is in the format of (start time, end time, label).

Split	Train	Val	Test
# video hours	262 (19)	87 (5)	84 (9)
# clips	$3.6k$ (164)	$1.2k$ (44)	$1.1k$ (69)
# queries	$13.6k$ (604)	$4.5k$ (164)	$4.4k$ (264)

Table 6. **Visual queries dataset statistics**. The numbers in the parantheses correspond to the subset of data used for 3D localization, where we focus on videos for which we have Matterport3D scans.

F.4 Data Analysis

We now overview the statistics of the annotations per query type.

Visual queries The VQ annotations consist of samples from a diverse set of scenarios and universities (see Figure 20 and 21). In total, 433 hours of videos are annotated with $22,602$ visual queries. These videos are sampled from 10 universities and consist of 54 scenarios. The statistics over the train/val/test splits are provided in Table 6. We ensured that the splits contain a disjoint set of videos. To look for possible biases in the data, we plot the distribution over three measures.

1) Query to response separation is the temporal distance (in frames) between the query frame and the end of the response track. This measures how far back in time an algorithm needs to search in order to find the query object.

2) Response track size measures the temporal length of the response track.

3) Response bbox position is the spatial start and end (x, y) coordinates for each bounding box in the response track. We normalize the coordinates by the image width and height to account for varying image sizes in the data. Each pixel within the bounding box contributes to an image heatmap that shows the frequency of each pixel belonging to a response track bounding box.

The analyses are shown in Figure 22. The query to response separation distances are fairly spread between 1 to 200 frames with a mode of $\sim$ 30 frames (see Figure 22, left). The response track sizes are well distributed between 1 to 40 frames with a mode of $\sim$ 8 frames (see Figure 22, center). The bounding boxes are near-uniformly distributed throughout the image, with very few bounding boxes annotated at the top 10% of the image (see Figure 22, right). Our analyses indicate that there may be a potential bias in the first two measures, while the bounding boxes positions are largely unbiased.

For the 3D localization task, we annotate a subset of 1,043 visual queries with 3D annotations. These comprise of 13 video hours associated with 4 scans from the University of Catania (UNICT).

Natural language queries As outlined in Table 7, the NLQ annotations are from 227 hours of video, with a total of $19.2K$ queries spanning the selected 13 query templates. The associated video clips come from 10 different universities with a total of 34 scenarios (with at least 1 hour of video annotated). Similar to other tasks within the episodic memory, we ensure that the train/val/test splits $(60\%, 20\%, 20\%)$ contain a disjoint set of video clips. We further analyze the data through: (a) Distribution over template queries, shown in Figure 24. The challenging *'Where is object X before/after event Y?'* is the most popular template with around

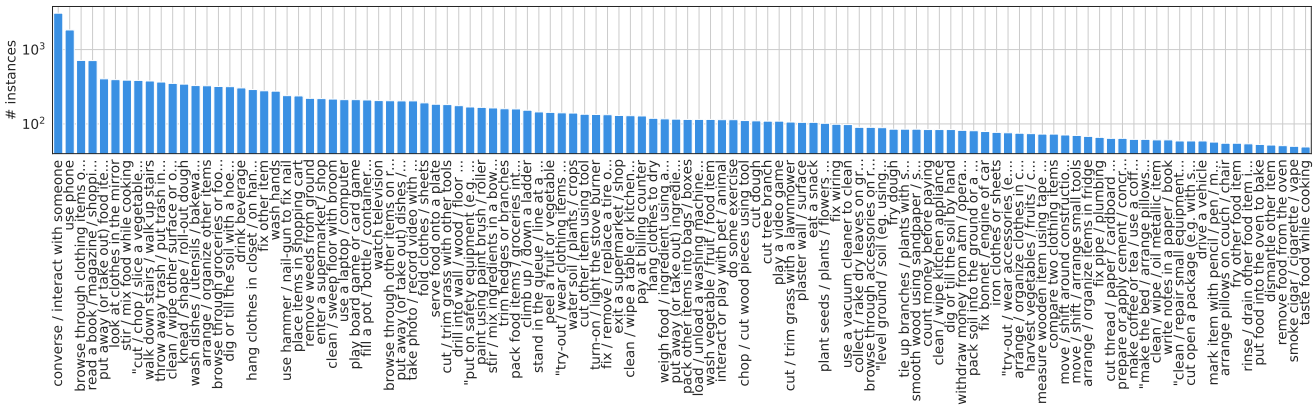


Figure 18. **Distribution of moments labels**. The figure shows the number of instances per category across 5 scenarios and 300 hours of data. All 110 categories are shown, sorted by frequency. The distribution is long tailed, with the smallest classes containing at least 50 instances. Note that these are only the Moments for Episodic Memory with temporal window annotations in the current release; Ego4D has many other scenarios and activities not reflected in this distribution.

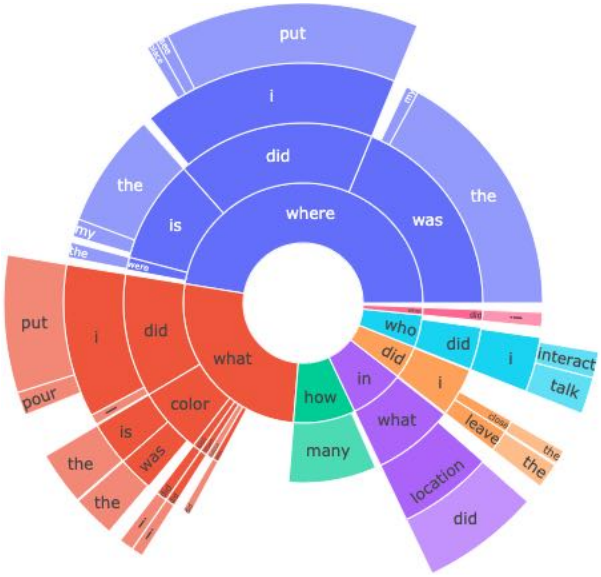


Figure 19. Distribution of query words in NLQ.

Split	Train	Val	Test
# video hours	136	45	46
# clips	$1.0k$	$0.3k$	$0.3k$
# queries	$11.3k$	$3.9k$	$4.0k$

Table 7. **NLQ dataset statistics** across the train/val/test splits.

Split	Train	Val	Test	Total
Video hours	194.9	68.5	62.9	326.4
# Video clips	1,486	521	481	2,488
# Instances	$13.6k$	$4.3k$	$4.3k$	$22.2k$

Table 8. **MQ dataset statistics** across the train/val/test splits.

$3K$ queries, with a reasonable distribution over other templates. Overall, the queries in NLQ have 8.3 ± 2.1 words in them. (b) Distribution of the response window length is shown in Figure 25. Typically, the windows are 9.3 ± 21.5 seconds long.

Most response windows are quite short compared to the full video clip, making the task a challenging "needle in the haystack" search problem. (c) Distribution of query words is shown in Figure 19. The branching off evidences the richness and diversity of the queries in NLQ.

Moments queries For MQ, similar to other tasks in episodic memory, we maintain a ratio of 6:2:2 among the train/val/test splits, which contains disjoint sets of video clips. To make sure there are enough samples in each category, we only keep categories that have at least 50 instances from the annotations and have instances in all train/val/test splits.

Consequently, the MQ dataset has 110 categories, spans a total 326.4 hours of videos, 2,488 video clips and $22.2k$ action instances. We summarize the statistics across the three splits in Table 8. We further explore the data through the following aspects. (a) The distribution of action duration is shown in Fig 26. We can see that most moments have very short duration. The majority of moments last less than 1 minute, and 22.4% actions have duration less than 3 seconds. Note that there is also a peak (2.6% instances) at the largest duration bin, where the actions almost cover the whole video clip. The average duration each instance is 45.2 seconds. (b) The distribution of different categories is shown in Fig 18. We notice that this is a long-tailed distribution, some categories (e.g., 'use phone', 'converse/interact with someone') with over 1000 instances and some categories with less than 100 instances. Each category has 205 instances on average. (c) The distribution of instance numbers in a video clip is shown in Fig 27. The majority of video clips have 1-20 moment instances, whereas very few can have as many as over 80 instances.

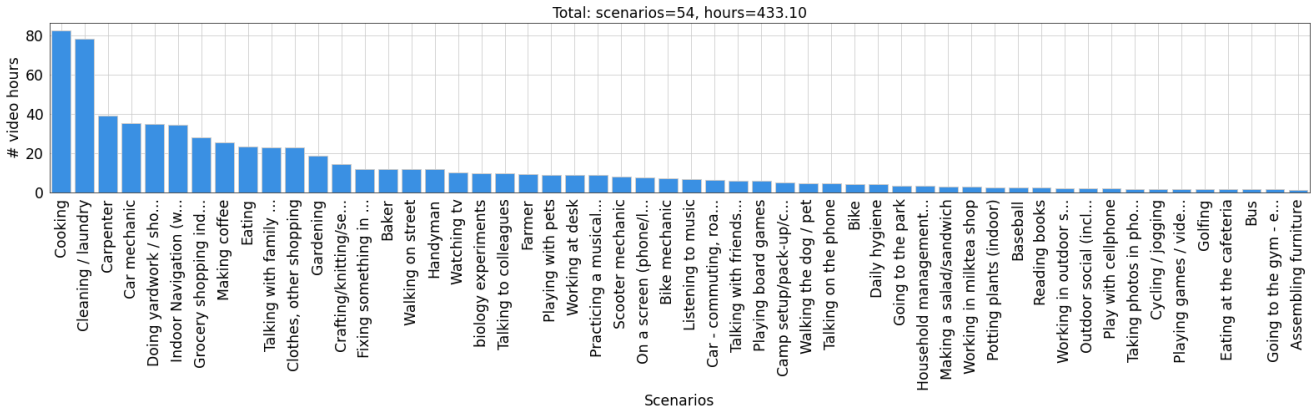


Figure 20. **Distribution over scenarios for visual queries.** The dataset contains a long-tail of scenarios. The plot title indicates the number of scenarios and the total video hours included in the dataset.

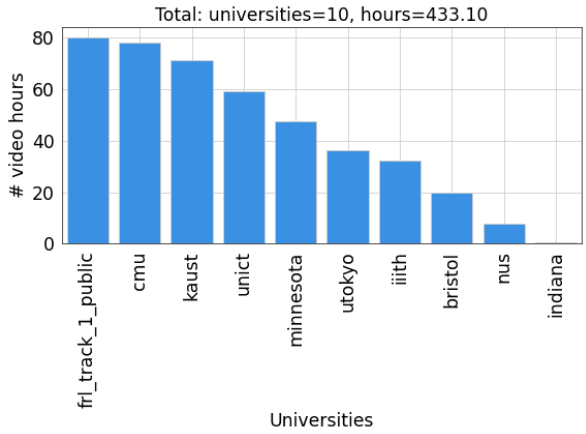


Figure 21. **Distribution over universities for visual queries.** The dataset contains annotations corresponding to videos from 10 universities. The plot title indicates the number of universities and the total video hours included in the dataset.

F.5 Evaluation measures

Next we detail the evaluation metrics for all three query types.

Visual queries We define the following localization metrics for the 2D localization task with top-1 retrieval.

Temporal AP (tAP) measures how closely the temporal extent of the prediction matches with the ground-truth response track. It is calculated as the average-precision of the predicted response track's temporal extent, and is based on the ActivityNet mAP metric [61]. We evaluate the tAP at 4 different tIoU thresholds $\{0.25, 0.50, 0.75, 0.95\}$, as well as their average value.

Spatio-temporal AP (stAP) measures how closely the spatio-temporal extent of the prediction matches the ground-truth response track. It is calculated as the average-precision of the predicted spatial-tube, and is based on the video-AP metric from [88]. We evaluate the stAP at 4 different stIoU thresholds $\{0.25, 0.50, 0.75, 0.95\}$, as well as their average value.

Success (Succ) measures whether the prediction has any overlap with the ground truth at all. It is calculated as the percentage of samples where the predicted response track has atleast 0.05 spatio-temporal IoU with the ground truth.

Recovery% (rec%) measures how much of the ground-truth response track is accurately recovered by the prediction. It is calculated as the % of frames in the response track where the predicted bounding box has at least 0.5 IoU with the ground truth. This is motivated by the tracking robustness metric from the VOT challenge [121].

Search efficiency (sEff) measures the efficiency of the algorithm searching for the query object. It is calculated as

$$\text{sEff} = 1 - \frac{n}{N} \tag{5}$$

where n is the number of video frames previewed by an algorithm to predict the response track, and N is the total number of frames in the video before the query was made (i.e., the search window). An algorithm that accesses every frame in the search window before localizing the query object gets 0.0 search efficiency. This "timeliness" metric is designed to encourage research on methods performing intelligent contextual-search.

We evaluate performance on the 3D VQ localization task using the root mean square error (RMSE) and the angular error metrics:

$$\text{RMSE} = \|t_s - \hat{t}_s\|_2 \tag{6}$$

$$\text{angular\_error} = \text{acos}\left(\frac{v_Q^T}{\|v_Q\|_2} \cdot \frac{\hat{v}_Q}{\|\hat{v}_Q\|_2}\right) \tag{7}$$

where t_s and $\hat{t}_s$ are the ground-truth and predicted object position in the scan coordinate system. v_Q and $\hat{v}_Q$ are the ground-truth and predicted 3D displacement vector in the query frame Q coordinate system. We also define a success metric leveraging the two annotations per query:

$$\text{succ} = \|c_m - \hat{t}_s\|_2 < 6 \times (\|c_1 - c_2\|_2 + \delta) \tag{8}$$

With $c1$ and $c2$ the centroids of the two bounding box annotations, c_m the mid-centroid between $c1$ and $c2$ and $\delta = \exp^{-m_{diag}}$, with m_{diag} the average diagonal length of the two boxes.

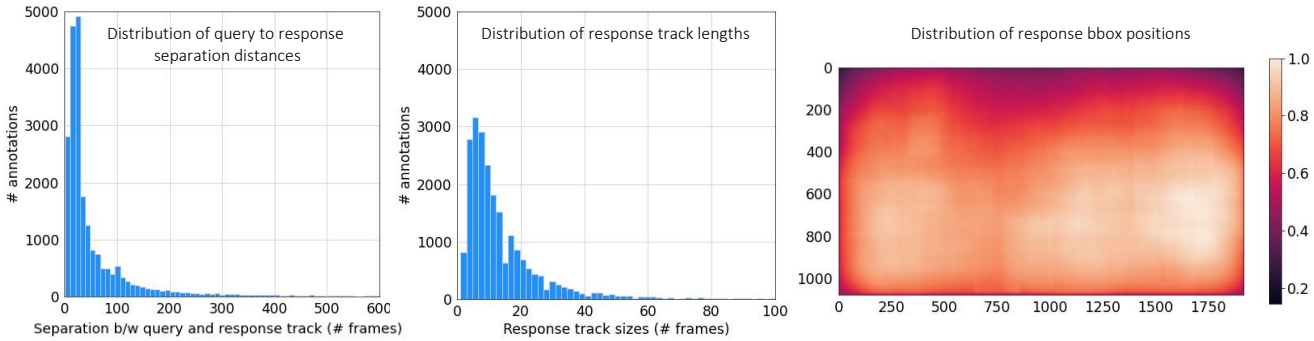


Figure 22. **Visual queries bias analysis.** We analyze the full VQ dataset for potential biases. **Left:** The plot shows the distribution of query to response separation distances in the VQ dataset. While the mode of the distribution is ~ 30 frames, we can see that separation distances are fairly spread between 1 to 200 frames. **Center:** The plot shows the distribution of response track sizes in the VQ dataset. While the mode of the distribution is ~ 8 frames, we can see that the response track sizes are well distributed between 1 to 40 frames. **Right:** The heatmap shows the normalized frequency of each pixel belonging to a response track bounding box. The bounding boxes near-uniformly distributed across most of the image.

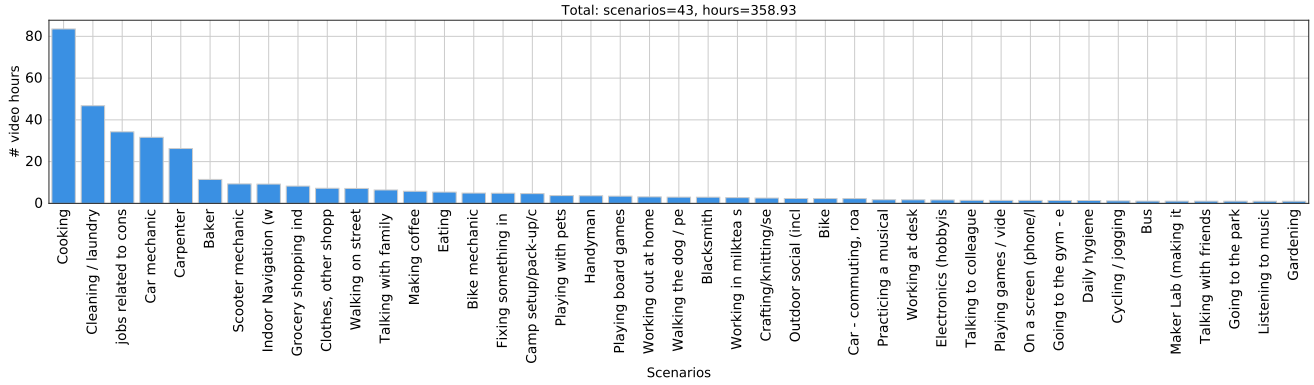


Figure 23. Distribution over scenarios for the NLQ annotations, indicating a long tail over scenarios. Note that the scenario labels are approximate and a single video can contain multiple scenario labels. For this plot, we equally divide the time across all the labelled scenarios.

Natural language queries Evaluation for NLQ is similar to existing video-language grounding problems. Following prior work [236], we use recall@k, IoU=m, where we select $k = \{1, 5\}$ and $m = \{0.3, 0.5\}$. This metric computes the percentage of times at least one of the top k predicted candidates have an intersection-over-union (IoU) of at least m. Note that we lean towards lower threshold values (m) as the average length of the window ($\sim 10s$) is much smaller than that of the video clip ($500s$), about 2% of the clip length.

Moments queries Considering that the moment queries task is related to the tasks of temporal action detection [61, 141, 229, 237] and video grounding [236], we adapt their respective metrics to moment queries.

Average Precision (AP) is a commonly adopted metric in temporal action detection. It measures how closely the temporal extent of the predictions matches the ground-truth action instances for each action category [61, 141, 229, 237] in terms of both precision and recall. The temporal intersection over union (tIoU) between a prediction and a ground-truth action instance is used to measure their distance. If the tIoU is higher than a threshold, the prediction is considered as true positive; otherwise, false positive. In representative temporal action detection datasets, such as ActivityNet [61], the mean AP (mAP) over all categories is computed given a tIoU threshold. Multiple tIoU thresholds are adopted, and the average mAP over all these tIoU thresholds is computed. For moment queries, we evaluate mAP at 5 different tIoU thresholds $\{0.1, 0.2, 0.3, 0.4, 0.5\}$, as well as their average value.

Recall@kx, tIoU=m, is a metric adapted from the metric recall@k, tIoU=m, used for NLQ. The metric recall@k, tIoU=m measures the percentage of the query sentences that have at least one prediction with a tIoU larger than the threshold m in the top-k results. In our moment queries case, since we might have more than one instance corresponding to a query moment category, we need to measure the percentage of all the correctly predicted instances that have at least one prediction with a tIoU larger than the threshold m in the top-k results of this instance. Considering that predictions are usually made based on a category not a specific instance, we modify the metric to be the following recall@kx, tIoU=m, where x stands for the number of instances for a query category in one video. This metric measures the percentage of all the correctly pre-

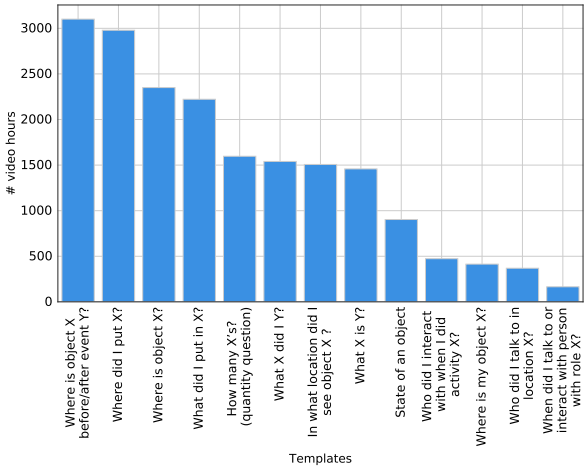


Figure 24. Distribution of queries over the corresponding templates across objects, place, and people categories (Tab.5). See text for more details.

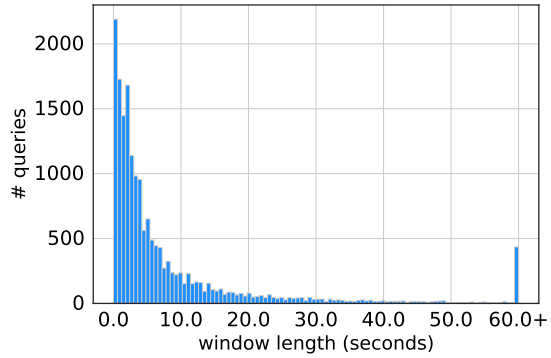


Figure 25. Distribution of response window length for NLQ. For the sake of brevity, we use the last bin to represent all windows longer than a minute. See text for more details.

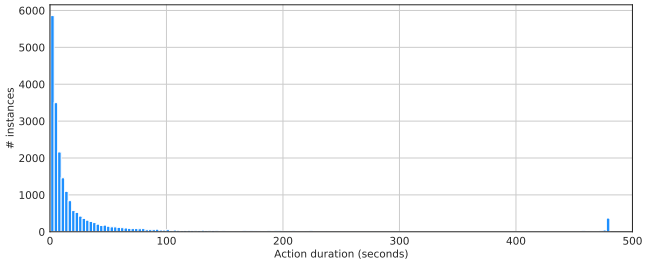


Figure 26. **Distribution of moment duration**.

dicted instances that have at least one prediction with a tIoU larger than the threshold m in the top-kx results of the action category. This metric has a similar idea to the multi-label metric proposed in [240] when dealing with multiple instances for a query. We use $k = 1, 2, 3$ and $m = 0.3, 0.5, 0.7$ in the metric. Compared to average precision, this metric only evaluates the recall for the query

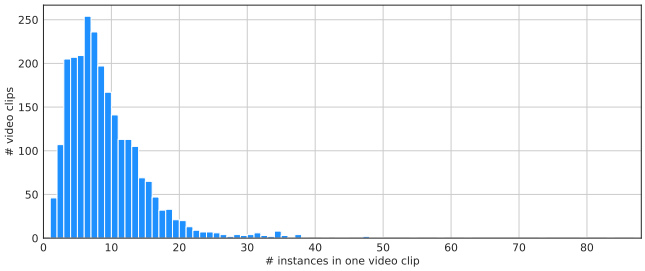


Figure 27. **Distribution of instance numbers in one video clip**.

categories, and does not penalize for false positive predictions given a category that has no instances in the video.

F.6 Baselines

We developed baseline models for each task. We designed these models to address our tasks, using state-of-the-art components where relevant. They represent a starting point upon which future work can build.

Visual queries 2D localization baseline

We treat visual queries with 2D localization (VQ2D) as a detection + tracking problem (see Figure 28). At a high level, our approach consists of three steps. First, we perform frame-level detection over the input video where we detect the presence of the query object in each frame using an object detection model (Figure 28 top). For each frame, we get the bounding box that is most similar to the visual crop and a score indicating its visual similarity. Second, we consider the sequence of per-frame similarity scores over the entire video and identify the most recent peak in these scores (Figure 28 bottom-left). Finally, we initialize a tracker at the video-frame corresponding to the peak detection, and track the query object on both forward and backward directions to recover the complete response track (Figure 28 bottom-right).

Step 1: Frame-level detection We propose Siam-RCNN, a Faster-RCNN [189] based approach to detect the query object in a given image. See Figure 28 top. Given a video frame at time t, a pre-trained Region Proposal Network (RPN) [189] with a Feature Pyramid Network (FPN) [142] backbone is used to generate bounding box proposals $\{b_1, \cdots, b_N\}$. The RoI-Align operation [94] is then used to extract visual features for each bounding box $\{\mathcal{F}(b_1), \cdots, \mathcal{F}(b_N)\}$. We use the same FPN backbone to extract features for the visual crop v. To detect the presence of the query object in frame t, each proposal feature $\mathcal{F}(b_i)$ is compared with the visual crop feature $\mathcal{F}(v)$ using a Siamese head $\mathcal{S}$ that predicts a 0-1 similarity score

$$s_i = \mathcal{S}(\mathcal{F}(b_i), \mathcal{F}(v)) \tag{9}$$

The Siamese network projects each proposal / visual-crop feature to a 1024-D feature vector using a convolutional projection module $\mathcal{P}$,

$$p_b = \mathcal{P}(\mathcal{F}(b_i));\ p_v = \mathcal{P}(\mathcal{F}(v)) \tag{10}$$

and predicts a 0 - 1 similarity score using a bilinear operation:

$$s_i = \sigma(p_b^T W p_v + b) \tag{11}$$

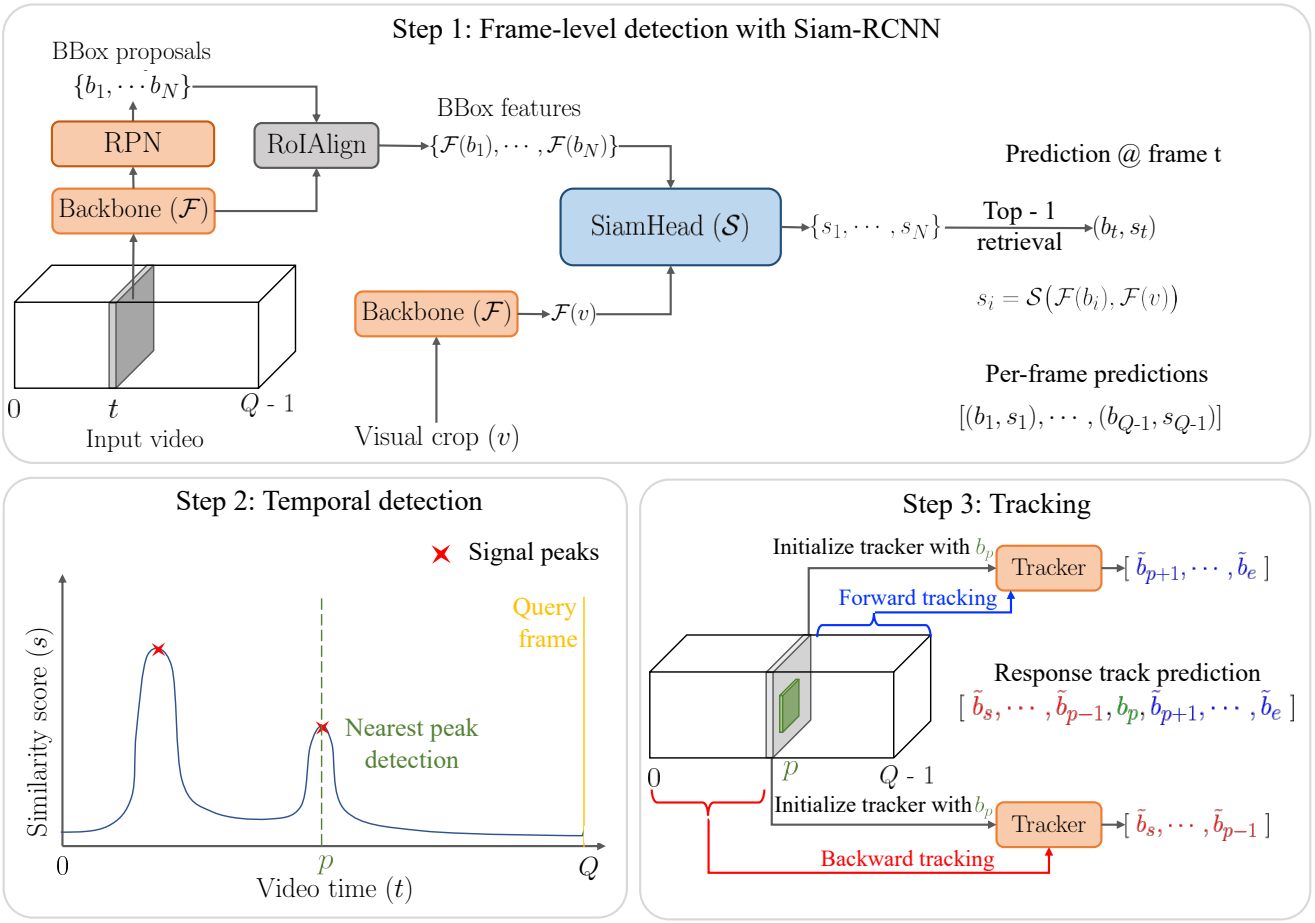


Figure 28. **Visual queries 2D localization baseline.** Our approach consists of three steps. **Step 1:** We perform frame-level detection for the entire input video to detect the presence of the query object (specified via the visual crop v). For each frame t, we extract the region proposals $\{b_1, \cdots, b_N\}$ using a region proposal network (RPN), and extract features for each proposal $\{\mathcal{F}(b_1), \cdots, \mathcal{F}(b_N)\}$. Each proposal feature is compared with the visual crop feature $\mathcal{F}(v)$ using a Siamese head $\mathcal{S}$, and the most similar proposal b_t is retrieved along with its score s_t. This process is repeated for all frames. **Step 2:** We treat the similarity scores $\boldsymbol{s} = \{s_1, \cdots, s_{q-1}\}$ as a temporal signal and perform temporal detection to obtain the 'most recent occurrence' of the query object. We detect the peaks (local maxima) in the signal and recover the peak p nearest to the query frame. **Step 3:** Given the detected peak p and its corresponding proposal b_p, we initialize two trackers with b_p and run them along the forward and backward directions to recover a contiguous track of the object, i.e., the response track prediction.

where σ is a sigmoid non-linearity. After computing the similarities to each bounding box proposal, the proposal b_t with the highest similarity score s_t for frame t can be obtained as follows:

$$b_t = \underset{b \in \{b_1, \cdots, b_N\}}{\arg\max} \{s_1, \cdots, s_N\} \tag{12}$$

$$s_t = \max\{s_1, \cdots, s_N\} \tag{13}$$

After repeating the above steps for all the video frames, we can obtain the final per-frame predictions as $[(b_1, s_1), \cdots, (b_{q-1}, s_{q-1})]$.

Step 2: Temporal detection So far, we used Siam-RCNN to get the most similar proposals and their similarity scores for every frame in the video. Next, the goal is to temporally detect the 'most recent occurrence' of the object in the video (see Figure 28 bottom-left). This is a challenging problem since our goal is not to identify the best detection of the object, but instead the most recent one, even if the similarity is not as high. To tackle this problem, we treat the per-frame similarity scores $\boldsymbol{s} = \{s_1, \cdots, s_{q-1}\}$ as a temporal signal, and use a signal peak detection approach to identify the salient peaks (a.k.a. local maxima) in $\boldsymbol{s}$. To avoid spurious peaks, we first smooth $\boldsymbol{s}$ using a median filter with a window size of 5.

$$\bar{\boldsymbol{s}} = \texttt{median\_filter}(\boldsymbol{s}) \tag{14}$$

$$p_1, \cdots, p_k = \texttt{find\_peaks}(\bar{\boldsymbol{s}}) \tag{15}$$

Depending on the video, the algorithm may return multiple peaks spread throughout the video (see signal peaks in Figure 28 bottom-right). Since our goal is to detect the most recent occurrence of the object, we select the peak p that is temporally nearest to the query frame.

Step 3: Tracking After temporal detection, we have identified a peak-frame p in the video which is estimated to have the most recent occurrence of the object. For this frame p, we can obtain the highest-scoring bounding box b_p from the per-frame detections in step 1. Note that this only represents one frame where the object most recently occurred. However, the task objective is to obtain the response track, i.e., the *contiguous set of all frames*, starting from when the object first entered the field-of-view until the object exits the field-of-view. See Figure 28 bottom-right. To compute the rest of the response track, we use b_p as a starting point, and run a single-object tracker forward and backward until the tracking fails (i.e., the object exits the field-of-view).

For both directions, we initialize the apperance model of the tracker using the proposal b_p. For the forward tracking, we run the tracker starting from frame $p+1$ to $q-1$ and obtain the tracked regions: $\boldsymbol{b_f} = [\bar{b}_{p+1}, \cdots, \bar{b}_e]$. For the backward tracking, we run the tracking starting from frame $p-1$ to 0 and obtain the tracked regions: $\boldsymbol{b_b} = [\bar{b}_s, \cdots, \bar{b}_{p-1}]$. We then concatenate $\boldsymbol{b_b}$, b_p, and $\boldsymbol{b_f}$ to obtain the complete response track prediction. We use the KYS tracker [22], which was shown to achieve state-of-the-art results for single-object tracking.

VQ2D baseline training setup We now discuss the training procedure for the VQ2D baseline. Each datapoint for the VQ2D task (defined on Ego4D videos) consists of the following: video V, visual crop image v, query frame number q, and response track boxes $r = \{r_s, r_{s+1}, \cdots, r_e\}$, where s and e are the start and end frames of r, and r_i is a bounding box defined on frame i of video V.

As a high-level overview, we initialize and freeze the backbone $\mathcal{F}$ and RPN using weights from an MS-COCO pre-trained Mask-RCNN model. We use the VQ2D annotations to train the SiamHead ($\mathcal{S}$). We initialize and freeze the KYS tracker using weights pre-trained on GOT-10k [99], LaSOT [62], and TrackingNet [162] datasets.

We next detail the training procedure for the SiamHead ($\mathcal{S}$). We use a similarity retrieval approach were the model is trained to predict high visual similarity between the visual crop v and positives, and low visual similarity between v and negatives. The loss function for $\mathcal{S}$ is a binary cross entropy loss defined over each (v, D_p, D_n) tuple (see Eqn. 16), where $D_p = \{p_i\}_{i=1}^{|D_p|}$ are positive detections, $D_n = \{n_j\}_{j=1}^{|D_n|}$ are negative detections, and $s_{x,v} = \mathcal{S}(\mathcal{F}(x), \mathcal{F}(v))$:

$$\mathcal{L}_{\mathcal{S}} = -\frac{1}{|D_p \cup D_n|}\left(\sum_{p \in D_p} \log(s_{p,v}) + \sum_{n \in D_n} \log(1 - s_{n,v})\right) \tag{16}$$

Both positives and negatives are defined based on proposals generated by the RPN. Given a visual crop v, a proposal p_i for $i \in (s, e)$ is a positive if the $\text{IoU}(p_i, r_i) \geq 0.5$, where r_i is the response track box in frame i. We remove all r_i which are too small, or have significantly different aspect ratios from the largest box in r since these typically correspond to obstructed views of the object. A proposal p_j is a negative if it satisfies any of the following two conditions:

1. $j \in (s, e)$ and $\text{IoU}(p_j, r_j) < 0.5$
2. p_j is sampled from another video.

We also found it beneficial to use hard-negative mining, where we initially sample a large number of negatives and then select the top-K negatives with the highest loss value.

We employ a few different augmentation strategies to artificially expand the dataset. First, we augment each data sample by replacing the visual crop v by a bounding box r_i from the response track. This works because the response track and the visual crop correspond to the same object. Next, we augment the visual crop v by applying random rotations between -120° to 120°. This exploits the fact that objects can have significant viewpoint variations in egocentric videos (unlike internet photos). Finally, we apply a random brightness augmentation to the video frames and the visual crop to simulate differing lighting.

Implementation details We train the SiamHead $\mathcal{S}$ using the Detectron2 library [225]. We use the default configuration file and make the following changes for our experiments. For each experiment, we use 8 GPUs, 64 visual crops per batch, and train for $300,000$ iterations with an initial learning rate of 0.02 followed by a $0.1\times$ decay after $200,000$ iterations. We extract backbone features from the "p3" layer of FPN. Based on validation performance, we use 6 positives and 64 negatives for each visual crop. Specifically, we sample 58 negatives per video frame which results in $58 \times 64 = 3712$ negatives per batch. For each visual crop, we sample the 64 hardest negatives out of 3712.

In the SiamHead $\mathcal{S}$ architecture, the projection module $\mathcal{P}$ consists of four residual blocks followed by average pooling, and a 2-layer multi-layer perceptron (MLP) with a hidden size of 1024-D and ReLU activation.

For signal peak detection, we utilize the `find_peaks` function from the scipy library[12] with the following hyperparameters selected through validation: distance = 25, width = 3, and prominence = 0.2.

Experimental results We evaluate the performance of multiple baselines on the VQ2D task in Tab. 9. The first column in the table shows the detection and tracking methods, and the second column shows the SiamHead projection architecture $\mathcal{P}$. In addition to the KYS tracker, we also experiment with a simple particle filter tracker (denoted 'PF') to assess the impact of the tracking quality. As an ablation of SiamRCNN, we replace the 4 residual blocks in the SiamHead projection module with a simple 3-layer CNN which has lower capacity with no residual connections (indicated by 'Simple').

We make several observations. When we use a simple projection model with a particle filter tracker, we already observe a good validation performance of 32.4% success, and 0.14 tAP_{25}. These can be attributed to using a strong proposal generator (RPN pre-trained on MS-COCO) and a learned siamese comparison model. Upon replacing the particle filter tracker with a SoTA KYS tracker [22], while the validation success rate remains similar at 33.0%, we observe significant gains (absolute) in all other metrics: 2% tAP, 2% stAP_{25}, and 14.3% recovery. This suggests that a good tracker is necessary to accurately capture the full response track after localizing a single frame within it. Finally, upon replacing the 'Simple' siamese projection with 4 residual blocks, we observe a significant

[12] Peak detection: https://docs.scipy.org/doc/scipy/reference/generated/scipy.signal.find_peaks.html

gains of 6.8% in success, 5% in tAP_{25}, 4% in $stAP_{25}$, and 5% in recovery %. This suggests that using a higher capacity model for the SiamHead is helpful for improving the per-frame detection performance for the VQ2D task. We observe similar trends on the test set. Please see Fig. 29 for qualitative examples of the model's predictions.

In all cases from Tab. 9, the search efficiency is 0% since the detectors are used on every frame in the search window. In Fig. 30 we experiment with two simple techniques for improving the search efficiency. The first approach uniformly subsamples k% of the frames in the search window (denoted as 'SS'). The second approach searches over only k% of the most recent frames in the search window, i.e., frames that are nearest to the query (denoted as 'N'). We consider 3 values of k in both cases: 10%, 25%, and 50%. Consider the results in Fig. 30. In both strategies, the search efficiency improves as we reduce k. The performance drops drastically for the 1st strategy where we subsample the search window, while it remains relatively stable for the second strategy where we preview a fraction of frames closest to the query. For example, we can achieve a search efficiency of 48.0% with only a $6-16$% relative drop in performance with $k = 50$% in the 2nd strategy. However, the performance drops significantly if we reduce k further. For example, we observe a reduction of $38-60$% for $k = 10$% with the 2nd strategy. This suggests that more intelligent methods that perform contextual search are needed to improve the search efficiency for VQ2D while maintaining good performance.

Visual queries 3D localization baseline

Next we describe the baseline for the visual query with 3D localization task. Recall the task definition: given a video, a query frame, and a visual crop of a target object, the goal is to output a 3D displacement vector from the camera center of the query frame to the center of the target object in 3D. The 3D position of the target object is defined at its most recent appearance in the video. Figure 31 shows a sample of the task.

Our baseline strategy has three steps. We first estimate the camera poses of the video. Then we retrieve the most recent instance of the target object in the video. Lastly, we estimate the depth of the detected object and retrieve its 3D position from the query frame.

Camera pose estimation The camera poses are estimated using a keypoint matching strategy along with a Perspective-n-Point (PnP) resolution approach. At a high level our approach consists of the following four steps. First we estimate the camera intrinsic parameters using Structure-from-Motion (SfM). Secondly, we extract and match keypoints from each frame in the video to keypoints extracted from the Matterport3D panoramas. Then, using the matched keypoints we set up and solve a PnP problem for each frame in the video to estimate the corresponding camera pose. Lastly, we refine the poses using temporal constraints.

Step 1: Camera intrinsics estimation We start by extracting a set of contiguous non-blurry frames from the video. In order to select non-blurry frames we compute the variance of the Laplacian on each image and select the ones with a value higher than a 100 threshold. We then select the largest contiguous set of non-blurry images. We cap the number of selected frames to 10 to limit the computational time of the SfM module. Once we have selected the images we run the automatic reconstruction module of COLMAP [196] to estimate the camera instrinsic parameters with a radial fisheye camera model.

Step 2: Keypoint extraction and matching We use SuperGlue [195] to extract and match keypoints. We first extract keypoints from the scan panoramas $\{k_{\{p,n\}}, p \in \mathcal{P}, n \in \mathcal{N}\}$ where $\mathcal{P}$ is the number of panoramas and $\mathcal{N}$ is the number of keypoints. The scan panoramas are generated using the Matterport SDK.[13] We render RGB and depth images at each scan position and sweep over pitch values $\in [-30, 30]$ with a step size of 5 deg. and yaw values $\in [-180, 180]$ with a step size of 15 deg. We generate on average 7K images per scan. Note that while we are not releasing the panoramas because of data anonymization concerns, we are providing the precomputed keypoints. Similarily, we extract keypoints from the video frames $\{k_{\{i,m\}}, i \in \mathcal{I}, m \in \mathcal{M}\}$ where $\mathcal{I}$ is the number of images in the video and $\mathcal{M}$ is the number of keypoints. Once the keypoints are extracted we loop through each frame $i \in \mathcal{I}$ in the video and match the extracted frame keypoints $\{k_{\{i,m\}}, m \in \mathcal{M}\}$ to all the panoramas keypoints $\{k_{\{p,n\}}, p \in \mathcal{P}, n \in \mathcal{N}\}$. We use the pretained models available[14] of SuperPoint [50] for keypoints and descriptors extraction and SuperGlue [195] for matching.

Step 3: PnP resolution We compute the camera pose for the video frames having at least 20 matched keypoints. We empirically find that a threshold of 20 provides a good tradeoff between the number of overall pose estimates and the quality of the estimations. The positions of the 3D keypoints are computed from a pinhole camera model of the Matterport camera using the rendered panorama depth, camera intrinsics, and camera pose. The positions of the 2D keypoints are directly extracted from the video frames pixels. We then use the OpenCV library to solve the PnP setup and estimate the camera pose from the matched pairs of 3D and 2D points and using the estimated camera intrinsic parameters. Using this method we can estimate the camera pose of roughly 2% of the total number of frames in the video. Next we incorporate temporal constraints to increase this number.

Step 4: Temporal constraints and final pose estimation To increase the number of estimates we refine the pose estimation pipeline by incorporating temporal constraints in an iterative procedure. We start by extracting and matching 2D keypoints from localized frames to non-localized ones in the video. This step is similar to the above Step 2; we use the same SuperGlue [195]. Using the matched keypoints and current estimated poses we triangulate new 3D keypoints for the non-localized images. We then solve a new PnP setup with the new keypoints. We apply this procedure iteratively until convergence. After refinement we achieve a performance of 15% of pose estimates of the total number of frames accross all video clips.

Camera pose estimation quality and sources of error We qualitatively evaluate the camera pose estimation pipeline by rendering the views in the 3D scans. Recall that the scans and videos have been recorded at different times and thus the scenes can contain large differences. Figure 32 shows camera poses estimates where left is the frame from the video, middle is the view from the scan, and right is the superposition. We see that even with large

[13] Matterport-SDK: https://matterport.github.io/showcase-sdk/sdk_intersection_inspector.html

[14] SuperGlue weights: https://github.com/magicleap/SuperGluePretrainedNetwork

Detector + Tracker	$\mathcal{P}$	Validation set						Test set					
		Succ	tAP	tAP_{25}	stAP	$stAP_{25}$	rec%	Succ	tAP	tAP_{25}	stAP	$stAP_{25}$	rec%
Siam-RCNN + PF	Simple	32.4	0.06	0.14	0.02	0.06	13.2	32.7	0.06	0.14	0.02	0.06	12.9
Siam-RCNN + KYS	Simple	33.0	0.08	0.15	0.03	0.08	27.2	33.4	0.09	0.16	0.03	0.08	26.9
Siam-RCNN + KYS	Residual	**39.8**	**0.12**	**0.20**	**0.04**	**0.12**	**32.2**	**41.6**	**0.12**	**0.21**	**0.05**	**0.13**	**34.0**

Table 9. **Visual queries 2D localization results.** We compare the performance of various baselines on the VQ2D validation and test datasets. Column 1 indicates the detector and tracker. Column 2 indicates the projection architecture used in case of the Siam-RCNN model.

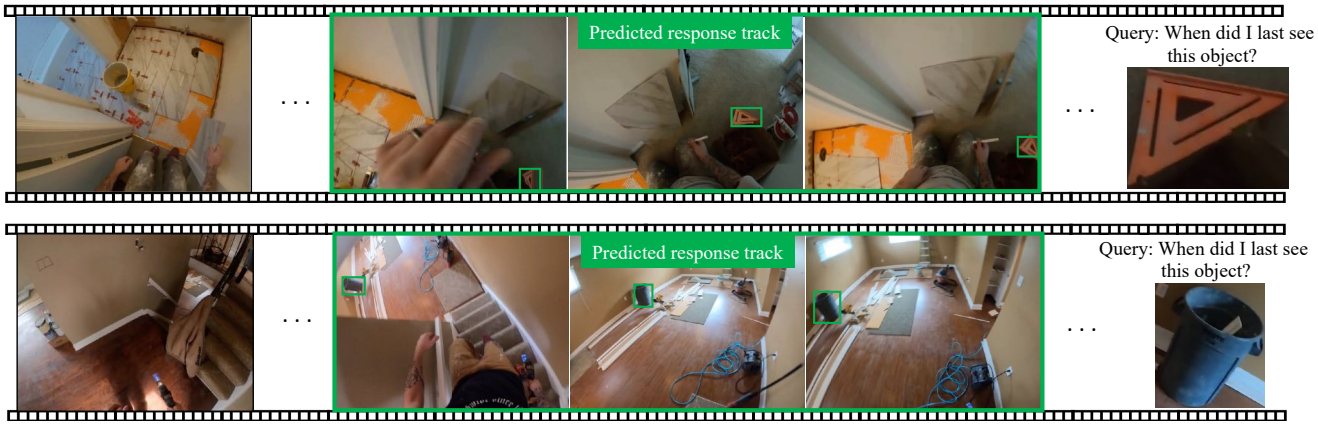


Figure 29. Qualitative examples for visual queries 2D localization. On each row, we show the visual crop of the query object on the right and the predicted response track in the center (3 uniformly samples images). The model was able to correctly localize the most recent occurrence of the object and accurately track it throughout the occurrence.

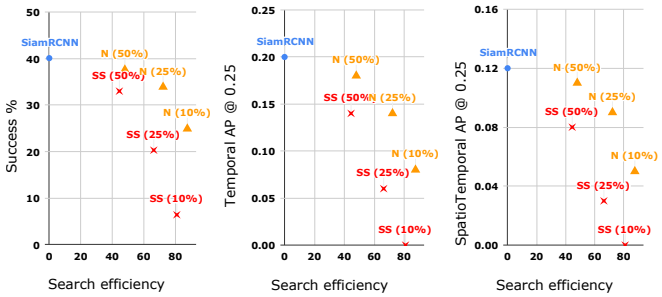


Figure 30. **Search efficiency for visual queries 2D localization.** We evaluate simple techniques for improving the search efficiency, and plot the corresponding VQ2D performance. The blue data point is the SiamRCNN performance when we preview the entire search window. The red data points are the SiamRCNN performance when we search over $k\%$ of the frames uniformly subsampled (SS) from the search window. The yellow data points are the SiamRCNN performance when we search over $k\%$ of the frames nearest (N) to the query (without any subsampling). The value of k is indicated above each data point.

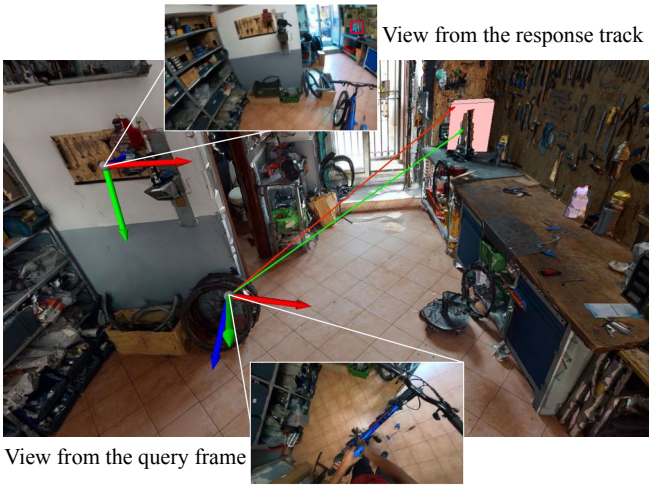


Figure 31. Visual queries 3D localization task demo. The top view is the view from the last frame of the response track with the target object annotated with a 2D red bounding box. The bottom view is the view from the query frame. The target object is annotated with a 3D red bounding box at the top right of the figure. The figure shows the ground-truth (green) and the predicted (red) 3D displacement vectors.

scene differences between the scan and video (e.g., the wheel in the middle example) the algorithm is capable of producing good pose estimates.

The remaining unlocalized frames are due to abrupt motion (lost track) and when the view is too close-up to the scene (not enough keypoints matched).

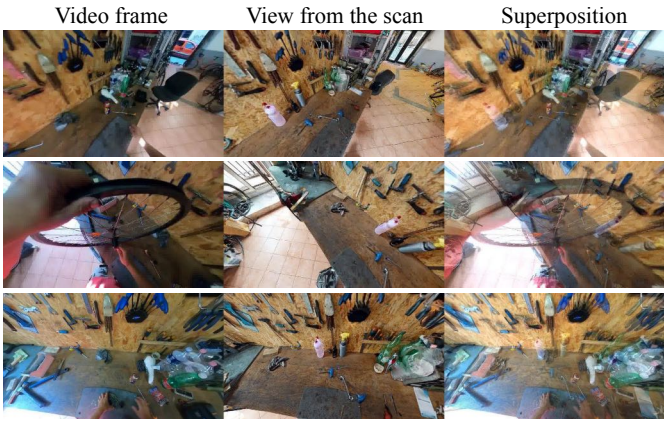


Figure 32. **Samples of camera pose estimation**. Left shows the frame from the egocentric video, middle has the view rendered from the estimated viewpoint in the scan and right is the superposition of both. We observe that even with big scene differences between the video and the scan (e.g., the wheel in the second row), the algorithm is able to accurately retrieve the camera pose.

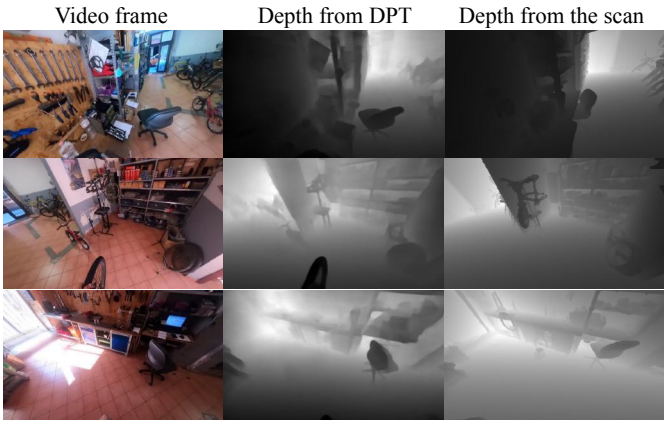


Figure 33. **Samples of depth estimation**. Left shows the frame from the egocentric video, middle has the estimated depth from DPT [185] and right has the depth from the scan rendered at the estimated viewpoint.

Target object retrieval We build our solution on top of the visual queries 2D localization baseline. The 2D localization baseline outputs a response track with 2D detections of the target object. Our baseline combines these 2D detections along with depth estimation and camera pose estimation to retrieve the 3D position of the object.

Depth estimation We estimate the depth of the most recent frame of the response track for which we have a pose estimate. We use the DPT network [185] with pretrained weights on NYU_v2 [202]. Figure 33 shows depth estimation results where left is the frame from the video, middle is the estimated depth, and right is the depth from the scan rendered at the estimated viewpoint (not available to the baseline model). Note that due to scene differences between the video and the scan, the two depths frames will differ in some region of the image. We then compute the depth value of the target centroid as the median of a square region centered at the 2D detection.

3D displacement vector reconstruction Given the estimated depth d of the object centroid c in frame f of the response track and the estimated camera instrisics K, we construct the 3D vector displacement $\hat{v}_f$ in the current frame f coordinate system using a pinhole camera model:

$$\hat{v}_f = \begin{bmatrix} x \\ y \\ z \end{bmatrix} = dK^{-1}c = dK^{-1} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \tag{17}$$

where u, v are the pixel indices of the centroid c in frame f. We then estimate the object centroid position $\hat{t}_s$ in the scan coordinate system:

$$\hat{t}_s = P_f^s \hat{v}_f \tag{18}$$

where P_f^s is the camera pose for the frame f. We further retrieve the displacement vector $\hat{v}_Q$ in the query frame Q coordinate system:

$$\hat{v}_Q = {P_Q^s}^{-1} \hat{t}_s \tag{19}$$

where P_Q^s is the camera pose of the query frame.

Experiments and results We compare the performance of multiple baselines along with ablation studies. We present the results in Table 10. Numbers are computed on the validation set (164 queries) of the VQ3D task. We report the query ratio QwP, for which we have camera pose estimates for the response track and query frame. Additionally, we report the success rate Succ^* which is the success metric computed only for queries with associated pose estimates.

Overall, we notice a low QwP ratio leading to a low success rate. These low metrics are due to a small number of camera pose estimates (15% overall). Nonetheless, we observe that the best VQ2D baseline method combined with the pretrained DPT [185] depth estimator yields the best performances in terms of $L2$ and success. These numbers tell that there are opportunities for enhancement in designing better camera pose estimators. Additionally, we perform ablation studies using the ground-truth response tracks and different depth estimators (random, from the scan, using DPT). For the random experiment we uniformly sample a depth value between 0.1 and 10 meters. From the ablation experiments we note that rendering the depth from the scan at the estimated viewpoint increases the performances compared to using DPT (lines 2 and 3). This suggests that there is also room for improvement in designing better depth estimators.

Natural language query baselines

Since the natural language queries can be seen as a language-grounding problem in a video, we adopt two prior methods in order to implement the baselines for this task.

(a) 2D Temporal Adjacent Networks (2D-TAN) [236]: We apply 2D-TAN with a sliding window method to implement the natural language query baseline. The goal of 2D-TAN is to answer where the semantically corresponding video moment is, given a

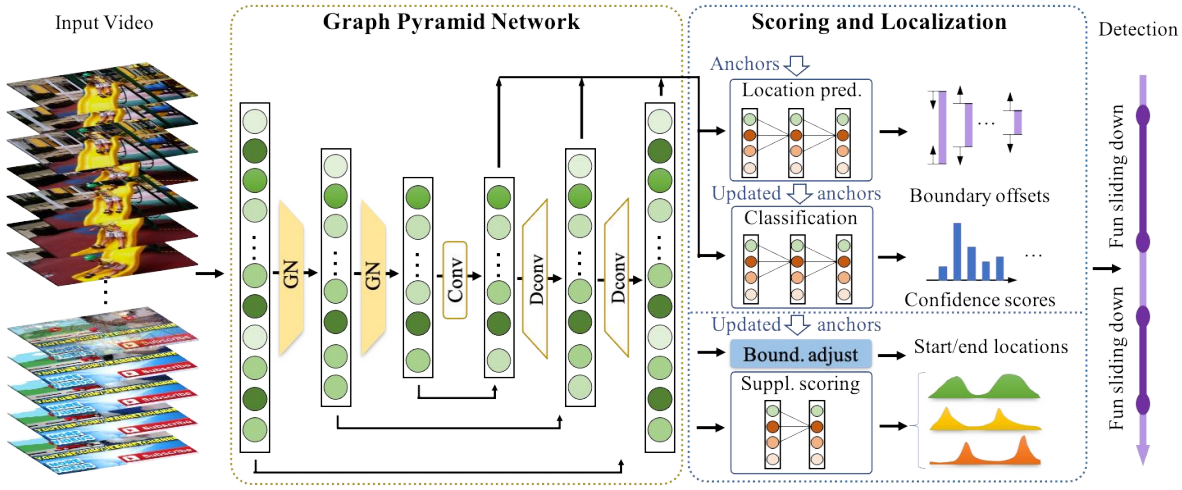


Figure 34. **Baseline model architectures: moment queries.** Its takes a video sequence and generates detected actions with start/end time, their categories, and confidence scores. It has two components: **graph pyramid network (GPN)**, and **scoring and localization (SoL)**. **GPN** is composed of multi-level encoder and decoder pyramids. The encoder aggregates features in different levels via a stack of graph networks (GN) (yellow trapezoid area; the decoder restores the temporal resolution and generates multi-level features for detection. **SoL** (blue dashed box) contains four modules, the top two predicting action scores and boundaries, the bottom two producing supplementary scores and adjusting boundaries. Figure is adapted from [237].

RT	depth	L2	angle	Succ*%	Succ%	QwP%
ground-truth	random	7.93	1.99	0.00	0.00	1.83
ground-truth	scan	2.92	1.10	76.47	1.22	1.83
ground-truth	DPT	3.33	1.15	76.47	1.22	1.83
Siam-RCNN + PF	DPT	6.53	1.64	25.00	0.61	**0.61**
Siam-RCNN + KYS (sim.)	DPT	**5.78**	**0.48**	**36.36**	0.61	0.61
Siam-RCNN + KYS (res.)	DPT	5.98	1.60	30.77	**1.22**	**1.83**

Table 10. **Visual queries 3D localization results.** We compare the performance of various baselines on the val set of the VQ3D task. Column 1 indicates the VQ2D network used to predict the response track (RT). The last metric QwP measures the query ratio for which we have pose estimation for the response track and the query frame. The $L2$ metric is expressed in meters and angles are in radians. The first three rows are ablation studies using the ground-truth response tracks and with depth estimated randomly, using the scan and via the DPT [185] network.

language query in an untrimmed video. The language query stems from one of the 13 template questions. The core idea of 2D-TAN is to consider adjacent moment candidates as the temporal context on a two-dimensional temporal map and retrieve the most relevant moment from the candidates. More concretely, 2D-TAN takes each moment candidate as one element in the 2D temporal map such that the adjacent moment candidates on the map can have much-overlapped content or share the same start or end time slot. It applies a convolutional neural network on the 2D map to predict the Intersection over Union of each moment candidate and the ground-truth moment. Please see [236] for more details.

Since the 2D-TAN enumerates all the possible combinations of start-end pairs, the $O(N^2)$ space complexity of the 2D map leads to a heavy model, especially when we require a precise moment boundary. To make 2D-TAN more appropriate to our problem, we further use a sliding window method on top of 2D-TAN. We break down the clip into a number of overlapping windows, where a window presents a small portion of the clip. The windows are taken as the input of the 2D-TAN model in both training and testing phases.

During the training of the 2D-TAN model, we use Ego4D's provided pre-extracted features for both the video clip and language query. The clip feature is from a SlowFast [71] network pretrained on Kinetics 400 dataset, and the language feature is a based on the BERT model [52]. The window duration is 40s, and stride is 20s in the sliding window method. Notably, we only use windows that contain or are next to a ground-truth moment in training, but we use all the windows in testing. We keep all the other hyper-parameters in 2D-TAN the same as its default except for tIoU threshold and learning rate. We decreased the tIoU threshold from 0.5 to 0.3 to enable more positive samples during training and empirically set the learning rate to 0.001. We train the model for 100 epochs and report the test set performance on the best checkpoint on the validation set. 2D-TAN gives top-1 and top-5 recalls of 5.80% and 13.90% at IoU=0.3, respectively. In addition, we also ablate the model to obtain performance by randomizing the video features ($-$visual) and textual features ($-$text) for NLQ in Tab. 11.

(b) Span-based Localization Network (VSLNet) [235]: Unlike traditional approaches in video natural language localization works, VSLNet treats the input untrimmed video as a text passage, and uses a span-based approach to identify the relevant sections semantically related to the given natural language query. At its core, VSLNet first encodes the natural language query and video features using a common, shared Transformer [215] network. Next, it uses

	Baseline	IoU=0.3 (%) r@1	IoU=0.3 (%) r@5	IoU=0.5 (%) r@1	IoU=0.5 (%) r@5
Val	2D-TAN [236]	5.04	**12.89**	2.02	5.88
Val	VSLNet [235]	**5.45**	10.74	**3.12**	**6.63**
Test	2D-TAN [236]	**5.80**	**13.90**	2.34	5.96
Test	−visual	2.29	6.77	1.32	3.46
Test	−text	3.46	10.13	1.78	4.38
Test	VSLNet [235]	5.47	11.21	**2.80**	**6.57**
Test	−visual	1.80	5.44	0.90	2.45
Test	−text	3.05	7.39	1.45	4.12

Table 11. Performance of the NLQ baselines on val and test splits.

the encoded query to then attend to the relevant parts of the video clip (akin to a text paragraph). The attended sections are further refined using a query-guided highlighting (QGH) strategy by extending the selection foreground of the video by a hyperparamter to capture more visual context. Please refer to [235] for more details on the motivation and architecture.

For our experiments, we maintain consistency with the other NLQ baselines and use pre-extracted features for both the video clip (SlowFast network [70]) and natural language query (BERT [52]). We use the implementation provided by the authors[15] with the following changes: (a) Set the video features size to 2304 dimensions to accommodate the features extracted from the SlowFast network, (b) Replace the text encoder to a frozen, pretrained BERT [52] model, (c) Set the internal dimension of the multimodal network to 128, and project the pre-trained BERT features from 768 to 128. We train the model for 200 epochs and pick the model with the best performance on val split. The corresponding test performance of this VSLNet model is reported in Tab. 11, along with visual and textual ablations.

Moments queries baseline

We formulate a moment queries baseline as a temporal action detection method [141, 229, 237], plus simple post-processing.

The MQ task only expects predictions for the query categories, whereas the temporal action detection task returns the predictions for all categories. Therefore, we can first use a temporal action detection method to predict for all categories, and only output the results corresponding to the query categories.

To predict all categories, we adopt a recent method VSGN [237], which was designed for temporal action detection in third-person videos. We use VSGN without the VSS component. Figure 34 illustrates the architecture. It takes a video $\mathcal{V}$ as input, extracts features for each snippet in the video using a network such as SlowFast [70], and feeds these features into a graph pyramid network. The graph pyramid network contains a encoder and a decoder, where the encoder is comprised of multiple levels of graph convolutional networks, and the decoder is comprised of multiple levels of de-convolutional networks. It is an anchor-based method that pre-defines temporal segments for each feature level as prediction reference. It predicts the scores and refines the locations of the anchors in two stages. In the first stage, it uses a region proposal network (RPN) from the decoder to predict class labels and regress boundaries for each anchor; in the second stage, it applies a boundary adjustment module to refine the boundary offsets based on the updated anchors from the first stage. It also has startness/endness predictions to provide auxiliary supervision and supplement scores for each predicted segment. Its output predictions are formulated as $\Phi = \{\phi_m = (t_{m,s}, t_{m,e}, c_m, s_m)\}_{m=1}^{M}$, where m is the number of predictions, $t_{m,s}$ and $t_{m,e}$ are start time and end time of the m^{th} prediction respectively, c_m is the predicted category, and s_m is the confidence score. For more details, please refer to [237].

Given a query category c, the retrieval results for the moment queries task are obtained as follows

$$\Phi_c = \{\phi_m = (t_{m,s}, t_{m,e}, c_m, s_m) \,|c_m = c, 1 \leq m \leq M)\}\,. \tag{20}$$

Implementation details For feature extraction, we use Ego4D's provided pre-extracted features using a SlowFast [70] network pre-trained on Kinects400 [108] at 1.87 features per second. The feature dimension is 2304.

Considering that the maximum clip length is 8 minutes, which has 897 features, we make the input length of our network 928 frames to cover the longest video clip. We have 5 levels in the graph pyramid network, each with temporal length 232, 116, 58, 29, and 14 respectively. We pre-define two base anchors of sizes 4 and 12 for Level 1 and increase the sizes by 2 for each deeper layer. We train for 30 epochs with a batch size 32 and learning rate 0.0001. In inference, we only apply per-category NMS with a confidence threshold 0.0005.

Experiments and results We show our baseline performance in terms of mAP in Table 12 and recall @ kx, tIoU=m in Table 13.

We provide further analysis on the average precision results using DETAD [9]. In Fig 35, we illustrate the proportion of each error type for the false positive predictions. It shows that both localization and classification are responsible for the false positive, improving either can increase the overall performance by a nontrivial amount. In Fig 36, we demonstrate the performance of different groups of moment instances based on moment duration and number of instances belonging to the same category per video clip. We notice that short moments tend to have low performance even though they are large in number. When there are 2-3 instances in one video, they are easiest to detect.

Table 12. **Moment queries results** on the validation set and the test set, measured by mAP (%) at different tIoU thresholds.

tIoU threshold	0.1	0.2	0.3	0.4	0.5	Average
Validation set	9.10	7.16	5.76	4.62	3.41	6.03
Test set	8.61	6.52	5.43	4.30	3.57	5.68

Discussion

Visual queries presents a novel and challenging task for object localization in egocentric videos. While our proposed baseline

[15] https://github.com/IsaacChanghau/VSLNet

Table 13. **Moment queries results on the validation set and the test set, measured by recall (R) @ kx, tIoU=m (%).**

m	0.3			0.5			0.7		
k	1	3	5	1	3	5	1	3	5
Validation Set	33.45	51.26	58.43	25.16	39.46	46.18	15.36	22.67	25.81
Test Set	33.56	52.23	59.79	24.25	39.22	46.22	14.83	23.15	26.28

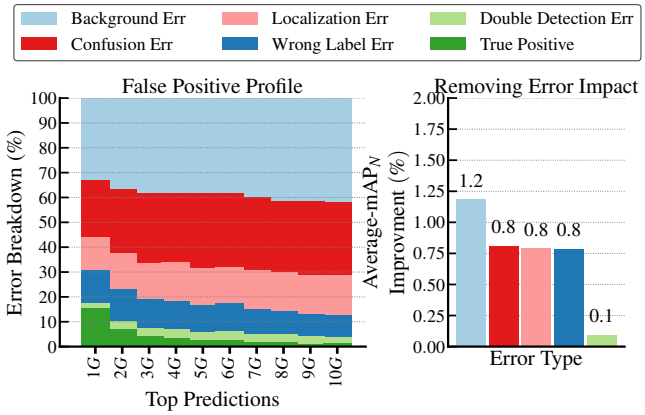


Figure 35. **Moment queries results: false positive analysis**. The error types are determined by the tIoU between ground-truth and predicted moments, as well as the correctness of the predicted labels, according to [9]. Background error: tIoU $< 1e^{-5}$; confusion error: $1e^{-5} <$ tIoU $< \alpha$, label is wrong; wrong label error: tIoU $>= \alpha$, label is wrong; localization error: $1e^{-5} <$ tIoU $< \alpha$, label is correct, where α refers to the tIoU thresholds $\{0.1, 0.2, 0.3, 0.4, 0.5\}$. 'G' refers to the number of ground-truth instances.

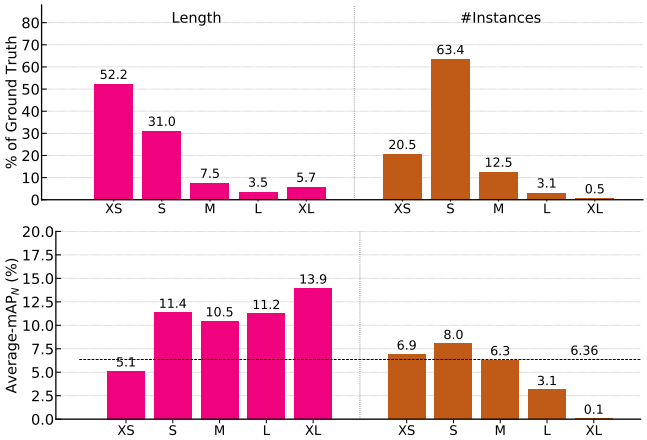


Figure 36. **Moment queries results: sensitivity analysis.** Top: Distribution of instance per action characteristic: length; # instances. Bottom: average mAP$_N$ (%) [9] in each characteristic bucket. The 'length' characteristic divides all moment instances 5 buckets based on the moments duration in seconds: XS (0, 10], S (10, 60], M (60, 180], L (180, 300], and XL (300, inf]. The '# instances' characteristic divides all moment instances into 5 buckets based on the number of instances belonging to the same category in one video clip: XS (0, 1], S (1, 3], M (3, 10], L (10, 20], and XL (20, inf].

achieves a reasonable success rate of 42.9%, it only achieves a localization performance of 0.13 tAP and 0.06 stAP. Furthermore, the best performance is achieved with 0% search efficiency, and naïve techniques to improve the search efficiency lead to drastic performance reductions. We hope that this task will spur future research into accurate and efficient techniques for object search.

Natural language queries is a challenging multimodal task that has wide applications in helping users search and retrieve relevant pieces of their episodic memory, thanks to the flexibility of the queries. The performance of the existing state-of-the-art video localization models highlights the needle-in-a-haystack nature of the task, due to shorter response windows of about $10s$ in a large video clip of 8 minutes. We hope that the NLQ dataset opens the door to future research that specializes in identifying and retrieving a large diversity of language queries in longer egocentric video clips, moving a step closer to augmenting a user's episodic memory.

Moment queries in egocentric videos is a challenging task due to the long-tailed distribution of categories and the large variation in moment duration. Our baseline achieves a reasonable result according to the metric recall @kx, tIoU=m, which evaluates the performance of each query category and does not require correct classification of all categories. In contrast, its average mAP score of 5.96% is low when all categories are evaluated. According to the false positive analysis in Fig 36, errors caused by wrong labels are significant. A more sophisticated classifier for all candidate moments can be explored in future work. In addition, as shown in Fig 36, the performance of short moments, which occupy a large proportion in the dataset, is not as good as that of long moments. Therefore, improving short moments will significantly improve the overall performance.

Contributions statement

Kristen Grauman led the Episodic Memory benchmark and paper writing, wrote annotation instructions, contributed to data selection and taxonomy formation, and co-advised the VQ baseline development. Bernard Ghanem co-led the Episodic Memory benchmark, managed baseline development and evaluation for the MQ and NLQ tasks, and contributed to the annotation instructions, data selection, and taxonomy formation for the MQ and NLQ datasets. Jackson Hamburger contributed to the development of the annotation instructions and taxonomies of the NLQ, VQ, and MQ datasets along with the design of the early VQ baselines.

Santhosh Kumar Ramakrishnan led VQ data selection, annotation, analysis and auditing, contributed to the formulation and annotation instructions of VQ, data selection, and implemented the VQ baseline. Vince Cartillier contributed to the VQ-3D formulation

and annotation instructions, led VQ-3D data selection, annotation, analysis and auditing, and implemented the VQ-3D baseline. Dhruv Batra co-mentored Vince Cartillier on developing baselines and provided guidance on 3D scans using Matterport. Hyun Soo Park contributed to 3D reconstruction of egocentric videos with respect to 3D Matterport scans. Tien Do developed algorithms to reconstruct 3D egocentric camera poses with respect to 3D Matterport scans.

James Hillis provided background knowledge on human episodic memory function and contributed to early discussions on benchmark definition and annotation. Satwik Kottur led the design of NLQ data selection and annotation instructions, contributed to the NLQ task formulation, coordinated NLQ data annotation and data analysis, implemented the VSLNet NLQ baseline, wrote part of the NLQ sections. Menmeng Xu designed and implemented the experiment pipeline for the NLQ task, implemented several NLQ methods, did NLQ result analysis and visualization, and wrote part of the NLQ sections. Michael Wray contributed to early formulation of the benchmark tasks, definitions of NLQ queries and annotation instructions, provided input for dataset construction and evaluation metrics, and helped in the creation of the MQ taxonomy.

Chen Zhao designed and implemented the MQ baseline, proposed and implemented the new metric for MQ, wrote the MQ sections, did MQ result analysis and visualization, contributed to the formulation, data selection and annotation instructions of MQ. Tushar Nagarajan contributed to the MQ formulation and annotation instructions, developed the MQ label taxonomy, and led the data selection and annotation of the MQ dataset. Merey Ramazanova managed the datasets for the experiments of MQ and NLQ baselines, and assisted with the taxonomy formation for the MQ baseline. Antonino Furnari provided keypoint feature extraction from the Matterport3D panoramas for the VQ3D baseline.

G. Hands and Objects Benchmark

This section details the Hands and Objects benchmark including definitions, annotations, baseline models and results.

G.1 Motivation

In a video of a human operating and manipulating an object with their hands, there may exist an *object state change*, *i.e.*, the point where the state of the objects being operated changes, either temporarily or permanently in a way that cannot be easily reversed. Examples of temporary state change include turning on a machine, while examples of permanent state changes include physical changes such as chopping a tomato into pieces and chemical changes such as mixing water and cement powder together to create a new composition of cement. Some examples are illustrated in Figure 37.

The concept of an object state change has been explored only in a limited manner in the video literature [8, 45, 69] and the characterization of state changes has depended on many brittle vision-based component technologies, making it difficult to analyze state changes at scale. Fortunately, in the last decade we have seen tremendous advances in computer vision algorithms for understanding both objects and hands. As a result, we believe that now it is time to investigate the idea of characterizing state changes at scale and in depth.

Why is recognizing the impact of agents on objects and environments so critical? We believe that understanding, recognizing, and replicating object state changes are an essential aspect of creating artificial intelligence (AI) systems. While current AI systems have the ability to replicate certain types of human actions such as assembling furniture [116] or cutting tomatoes [200], most systems do not possess a general understanding of how the environment and the objects can be transformed as a result of interaction. Understanding the impact of interactions on objects and the environment is an important aspect of reasoning and can help AI systems perform more advanced tasks. For example, understanding the impact of interactions on the environment can help AI systems relate multiple ways to achieve the same change, discover efficient methods for achieving goal states, recognize the completion/incompletion of goals [58, 97], recover from failure, and learn from mistakes.

In egocentric videos specifically, the object state changes offer rich and important information that are related to many other problems. For example, the object undergoing state change in an egocentric video can imply human-centric information such as human activity and intention. Moreover, the state change of an object shown provides cues about human-specific affordance and actionable information of an object or tool, which cannot be easily inferred from static images. Additionally, a joint understanding of human hands and the objects undergoing state change can benefit applications that require rich human demonstrations, such as robotic manipulation.

Defining Object State Changes: This benchmark focuses on identifying and localizing the state change of an object in an egocentric video. Specifically, a object state change can be represented by the three aspects in the video: temporal, spatial, and semantic.

Temporal: An object state change can be represented by three distinct temporal points in the video. *(1) Point-of-no-return:* The point-of-no-return (PNR) is the frame I_{pnr} in a video that identifies the beginning of an object state change that cannot be easily reversed. *(2) Pre-condition:* The pre-condition is defined as some frame I_{pre} that marks a moment prior to the state-change in which the related objects were visible within the field of view of the camera. *(3) Post-condition:* The post-condition is some frame I_{post} at which the completion of the state change is visible after the point-of-no-return. These three frames mark the distinct temporal stages of the object state change: before and after the change, respectively. This proposal matches the Rubicon Boundaries proposed in [160].

Figure 37. Examples of object state change. (a) State change through construction: attaching to two metal plates results in a new object. (b) State change through physical change: cutting a piece of wood results in two smaller pieces of wood. (c) State change through chemical reaction: combining two objects, water and cement powder, results in a new object, cement.

Spatial: An object state change can be represented by the bounding box of the object at the PNR, pre-condition and post-condition, along with any tools involved in performing the state change. Tools offer extended capabilities of the actor's hand, such as using an electric saw to cut a piece of wood in half. These bounding boxes represent the spatial dimensions of hands, tools and the objects undergoing the state change.

Semantic: We represent an object state change through the human action (verb), the object identity (noun) and the type of state change applied. The same state change can be performed on different objects using different tools. For example, cutting a piece of wood with electric saw and cutting a piece of paper with scissors are different interactions with different objects and different tools but they both result in the same object state change of *being cut*.

G.2 Related Work

Object State Changes: Existing approaches for modeling object states and/or their changes can be categorized into two research lines. The first deals with collections of images. A representative dataset for this purpose is the MIT States dataset [103]. By considering object states as object attributes (e.g. burnt, sliced), this line of work studies attribute-object composition, e.g. composition

with context [158], modeling attributes as operators [164], and an architecture for compositional reasoning [182].

The second research line deals with video and views an action as a state transformation over time. One direction is the discovery of object states and/or manipulating actions, *e.g.* in egocentric [45, 69] and instructional videos [8]. Fathi et al. [69] explore object state detection in video using a weakly supervised approach. Another direction is the modeling of state transitions. Zhou et al. [244] study temporal transformations of a single object state in time-lapse videos. Wang et al. [223] propose to model state transformations in a high-level feature space with Siamese networks. Doughty et al. [55] leverage natural language and treat adverbs as modifiers for state transformations. In terms of applications, Chang et al. [30] show state transformations can be utilized for procedure planning.

Human Hand Action Datasets: Several video datasets have been proposed for human hand action recognition. The Yale human grasping dataset [25] focuses on human grasping behavior and consists of 27.7 hours of annotated videos. The Something-Something dataset [90] consists of 220,847 short videos annotated with 174 categories of general hand-object interactions. The Jester dataset [214] provides 148,092 short videos in 27 hand gesture types. Wang et al. [220] construct a synthetic video dataset of human-object interaction through rendering hand and object CAD models. The recent Human Hands dataset [198] annotates 100K single frames from web-based videos, focusing on hand interactions and the offset between the hand and the interacting object during interaction.

Several egocentric video datasets capture daily living activities by people [43, 130, 136, 180, 201, 210]. In the Activities of Daily Living Dataset (ADL), subjects wear chest-mounted cameras and perform unscripted activities at home, with a total of 10 hours of video from 20 participants; the target task is activity recognition [180]. In the UT-Egocentric dataset (UT-Ego), subjects wear a head-mounted camera and perform long unscripted activities inside and outside of the home, with a total of 17 hours from 4 subjects (4-5 hours of continuous capture for each person); the target task is video summarization [130]. The UT Egocentric Engagement (UT EE) dataset consists of 14 hours of head-mounted camera video captured in public spaces like museums, malls, and grocery stores, and is annotated for moments of engagement by the camera wearer with the environment. In the EGTEA+ dataset, 32 subjects wearing head-mounted cameras in a single kitchen environment capture 28 hours of video; the task is to recognize 44 meal preparation activities [136]. The EPIC-KITCHENS dataset consists of 100 hours of kitchen activities recorded in 45 unique environments, with a total of 89,977 different object interactions across 97 verb and 330 noun classes; the task is to recognize objects and activities and anticipate interactions in the next moment of video [43]. The Charades-Ego dataset consists of 34 hours of video from 71 participants, with both first- and third-person paired instances labeled for 156 actions [201].

G.3 Benchmark Definitions

We now define the three tasks that comprise the Hands and Objects benchmark. The three tasks correspond to the three aspects of object state changes described above, namely, the temporal, spatial and semantic aspects of a state change.

(1) PNR Temporal Localization. The goal of Point-of-no-return (PNR) Temporal Localization is to predict I_{pnr}. One possible formulation is to view this problem as a per-frame classification problem, predicting the Point-of-no-return frame within a short video clip. The performance is evaluated only on the videos that contain object state change, and is measured by the absolute temporal error of I_{pnr} prediction in seconds.

The PNR was first discussed by P. Gollwitzer in his well-cited handbook of behavior [89]. Specifically, the book proposes the Rubicon Model of Action Phases, focusing on hand-object interaction. Action phases are delimited by three transition points: initiation of prior motion, PNR, and goal achievement. This was later experimentally assessed by our previous work [160], where PNR annotations were acquired for three egocentric datasets, demonstrating increased accuracy of annotations (see Fig. 10 in [160]) and improved robustness in training models (see Sec. 5 in [160]). Below, we find PNR closely aligns with the narration timestamps that we independently collected, suggesting PNR is a natural time point for human understanding (and thus narration) of the interaction.

(2) State Change Object Detection. We define a State Change Object as the object that is manipulated by a person and undergoes a change in its state. The goal of this task is to predict the 2D bounding boxes of the State Change Object in Point-of-no-return frame I_{pnr} given three frames: Pre-condition I_{pre}, Point-of-no-return I_{pnr}, and Post-condition I_{post}. We expect that a good solution to this task would incorporate the visual information before and after state change to detect the State Change Object. The detection performance is evaluated on the bounding boxes estimated in the Point-of-no-return frame I_{pnr} and measured by Average Precision (AP).

(3) Object State Change Classification. The task of Object State Change Classification classifies a short video clip to a state change type. With N object state change types defined, object state change classification is essentially an $(N + 1)$-way classification problem, where the one additional category is "without state change." Object State Change Classification is evaluated by classification accuracy.

G.4 Data Selection

Next we describe our data selection procedure and annotation pipeline, and we present the analysis of the data for the object state change benchmark. We begin by describing our procedure for selecting the subset of data to annotate for this benchmark.

We start with a large pool of videos annotated with high-level scenario labels (*e.g.*, gardening, cooking, landscaping, *etc.*) and narrations. We assess each scenario on the scale of 0 to 3 based on how likely it is to contain hand-object interactions (*e.g.*, 0 for "watching tv", 3 for "carpentery", *etc.*). We then sample data to annotate following the resulting scenario distribution. Given a scenario and a target number of hours, we sample clips randomly in a hierarchical fashion: we first sample a participant, then a video, and finally a 5 minute clip from the video. If the video is shorter than 5 min we take the whole video. For each scenario, we balance the data across universities to maximize geographic diversity. The resulting scenario and university distributions are

shown in Figure 38. In total, our dataset has 120 hours representing 53 scenarios, 7 universities, and 406 participants.

G.5 Data Annotation

We annotate hand-object interactions corresponding to each narration within the selected 5 minute clips. We use the taxonomy from Section D.3 for semantic verb and noun labeling. The annotation pipeline consists of three sequential stages: critical frame labeling, pre-period labeling, and post-period labeling.

Critical frames. Given a narration, we create an 8 second video snippet centered at the narration time point and present it to the annotators. We ask the annotators to first read the narration and select a corresponding verb from the taxonomy. The annotators can then play the video back and forth to select three critical frames in time: PNR, PRE, and POST. We ask the annotators to start with the PNR frame that identifies the beginning of the state change. This frame is less ambiguous and helps provide the context for the interaction. We then ask the annotators to label a frame prior to the state change (PRE) and a frame after the completion of the state change (POST). Note that the PRE and POST frames are not uniquely defined. We let the annotators pick any, as long as the relevant objects are *fully* visible within the field of view of the camera.

Pre period. Next, we label bounding boxes for the hands, tools, and objects, as well as the category names for the tools and objects. We do this in two steps. First we label the frames in the pre period, starting at PNR and going backward to the pre frame. The video frames are reversed and the annotators can play the video. We find that it is easier to start from the PNR frame since the hands and objects are clearly visible. To speed up hand box labeling, we initialize the hand boxes with a pre-trained object detector [198] and ask the annotators to correct these.

Post period. Finally, we ask the annotators to label spatial annotations and categories for the post frame. As before, we first present the annotators with the PNR frame. Note that in this case the PNR frame is already labeled which helps identify the hands and objects to label in the post frame.

G.6 Data Analysis

Finally, we present the analysis of our annotations.

Critical frames. In Figure 40 we show the temporal distribution of critical frames within the 8 second hand-object interaction snippets. First, we observe that the PNR frame distribution is centered around the middle of the 8 second snippet. Interestingly, this closely aligns with the narration point (4s mark). Next, we see that most of the pre and post frames come shortly before and after the PNR frame, respectively, highlighting the quick nature of these state changes, and thus the challenge in this benchmark. We also notice two additional modes for pre and post frames that come at the start and the end of the 8s interval, respectively. These correspond to long repetitive actions that start before or continue past the video snippet (*e.g.*, knitting).

Hands and objects. Our benchmark contains a large number of hands and objects annotated with bounding boxes. In total, we have ∼825K bounding boxes, including ∼245K for left hand, ∼260K for right hand, ∼280K for objects, and ∼40K for tools. In Figure 41 and Figure 42, we show the distributions of box sizes and locations, respectively. We observe that our data contains hands and objects at a variety of sizes and locations.

Actions. One of the features of our benchmark is the diversity of interactions. We focus on low-level atomic actions rather than high-level actions. We show the distribution of verbs (Figure 39, left) and nouns (Figure 39, right). We see that we have a large number of verbs corresponding to common manipulation actions (*e.g.*, put, take) and a natural long tail. The object distribution follows the same general trend. We note that our objects are common daily objects that are not typically present in object detection datasets (*e.g.*, 442 out of our 478 object categories cover categories beyond the 80 COCO [143] categories).

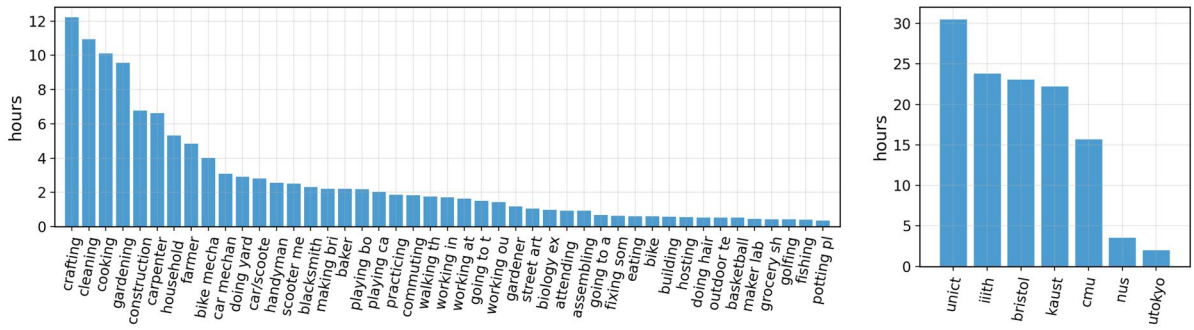
G.7 Baselines: Object State Change Classification and PNR Temporal Localization

We present the implementation of several baseline methods for the Object State Change Classification and PNR Temporal Localization tasks. Among the implemented baseline models, in general there are one or two types of output network heads: a classification head for the video clip used for state change classification, and/or a per-frame classification head for temporal localization. One can choose to train two models separately, or use the same backbone model but two network output heads and train the joint model with a multi-task loss function. The following baseline methods includes both types of model designs:

I3D ResNet-50. We use I3D [29] with ResNet-50 [95] as backbone architecture of the model for both the Object State Change Classification and the PNR Temporal Localization tasks. The ResNet backbone is followed by two network output heads: a state change classification head and a PNR temporal localization head. The state change classification head is produced by global average pooling on the entire spatiotemporal feature tensor followed by a classification layer. The PNR temporal localization head is produced by per-frame average pooling followed by a classification layer. The overall training loss of the model is the combination of the loss of two heads which are both cross-entropy loss for classification.

Boundary Matching Network (BMN). We use BMN [140] as a baseline for the PNR Temporal Localization task. BMN is a temporal segment detection method based on confidence prediction of dense temporal segment proposals. We view the start of the video as the start of the temporal segment and Point-of-no-return I_{pnr} as the end of the temporal segment, so we can convert the problem of localizing Point-of-no-return I_{pnr} to the problem of detecting the temporal segment. In our implementation, BMN uses ResNet as the backbone model. Furthermore, BMN is only used for the PNR temporal localization task.

SlowFast + Perceiver. We implement a baseline model whose architecture consists of SlowFast [70] and Perceiver [105] for both object state change classification and PNR temporal localization. SlowFast acts as the video deep feature extractor. The features are then passed to a Perceiver model. Similar to the previous BMN model, the SlowFast + Perciever model is only trained for temporal



(a) **Scenarios** (b) **Universities**

Figure 38. **Number of hours.** We show the distribution of the number of hours across scenarios (left) and universities (right)

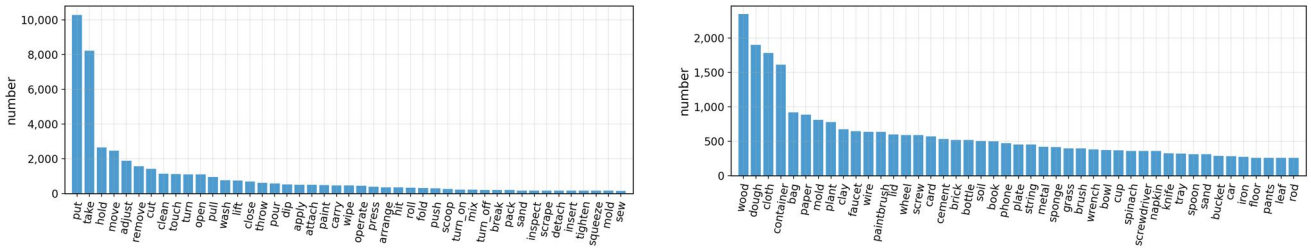


Figure 39. **Labeled actions.** Distribution of verbs (left) and nouns (right) in annotated action instances. Top 45 verbs and nouns are shown for clarity. See Section D.3 for more details.

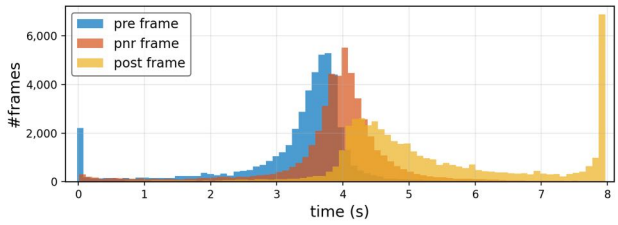


Figure 40. **Critical frames.** Distribution of critical frame times. Shown relative to the 8s hand-object interaction snippet.

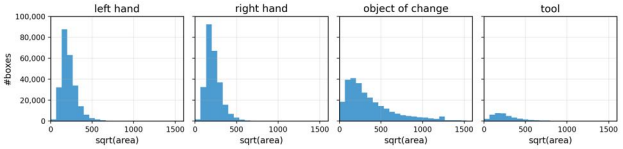


Figure 41. **Hand and object sizes.** Distribution of bounding box sizes. Shown in terms of the square root of the box areas.

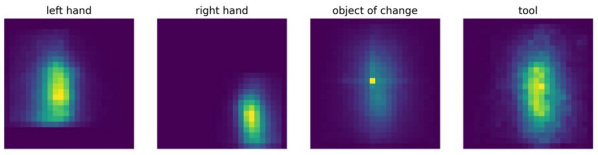


Figure 42. **Hand and object locations.** Distribution of bounding box centers. Shown in normalized image coordinates.

Table 14. Number of positive and negative video clips of object state change in train, validation and test splits.

Split	Positive	Negative	Total
Train	20,041	21,044	41,085
Val	13,628	14,720	28,348
Test	13,561	14,870	28,431

localization task. The training loss of the model is the cross-entropy loss for per-frame classification.

Bi-directional LSTM. We implement a Bi-directional LSTM model [91] for both the object state change classification and PNR temporal localization. We first pass individual frames to a ResNet model [95] to extract deep features. The sequence of per-frame features is then passed to the Bi-directional LSTM as input, with the output sent to both the per-frame classification head and the whole-sequence classification head. The overall training loss of the model is the combination of the loss of two heads which are both cross-entropy loss for classification.

For the object state change classification tasks, in the current version we focus on the two-way classification problem of whether there is a object state change in the egocentric video. In Table 14,

Table 15. Results of State Change Classification accuracy (%).

Baseline	Val	Test
Always Positive	48.1	47.7
Bi-directional LSTM [91]	65.3	63.8
I3D ResNet-50 [29]	68.7	67.6

Table 16. Results of Point-of-no-return temporal localization error (seconds).

Baseline	Val	Test
Always Center Frame	1.032	1.056
BMN [140]	0.780	0.805
I3D ResNet-50 [29]	0.739	0.755
Bi-directional LSTM [91]	0.790	0.759
SlowFast [70] + Perceiver [105]	0.804	0.828

we illustrate the number of positive video clips that contains an object state change and the number of negative video clips that do not contain object state change in the train/val/test splits. In all three splits, the positive and negative clips are balanced in number.

Besides the above learnable baselines, for object state change classification, we also present the result of the naive baseline of always predicting the positive category as the prediction. For the PNR temporal localization task, we additionally present the result of the naive baseline of always selecting the center frame of the trimmed video as the PNR frame, given the possible centre bias of the data.

The results for object state change classification task are illustrated in Table 15. The naive baseline of always positive prediction yields state change classification accuracy of close to 50%. All the learnable baselines outperform the naive baseline and achieve accuracy of more than 60% while Bi-directional LSTM baseline achieves the best performance. This shows that the learnable baselines can learn meaningful information about object state change, though there is clearly still space for improvement. One challenge in this task is that there is very large variance in term of the types of object state changes and objects contained in the videos.

The results for the PNR temporal localization task are illustrated in Table 16. The naive baseline of always predicting the center frame yields a temporal localization error of around 1.1 seconds. Other learnable baselines can achieve better temporal localization error of around 0.85 seconds or less which shows the baseline models can learn meaningful information for temporal localization of object state change. Note that the SlowFast + Perceiver model achieves the best temporal localization performance of 0.425 seconds on validation set and 0.489 seconds on test set, which highlights the necessity of using attention-based mechanism to model the change of object state. One challenge for this task is that in some actions, *e.g.*, cutting a piece of paper with scissors, the state change of an object does not necessarily cause significant change of visual appearance and therefore it is difficult to localize the PNR.

Table 17. Number of State Change Object and hand bounding boxes in train, validation and test splits.

Split	State Change Object	Hand
Train	19,347	33,254
Val	12,912	22,098
Test	13,118	22,576

G.8 Baselines: State Change Object Detection

While we expect that new methods developed for the tasks of state change object detection will utilize all three input frames (pre, PNR, post), in this initial stage of the benchmark, we only evaluate single-frame detection baselines, where only the PNR frame I_{pnr} is used as input. We limited our input as many methods for object detection are primarily designed to work with a single image.

We present the implementation of several baseline methods for the state change object detection task. In general, the baseline models for the task can be categorized into two types: (1) directly detecting the bounding box of the state change object including Faster-RCNN [190], CenterNet [241], and DETR [27], and (2) detecting hand bounding boxes first then predict state change object bounding boxes given the hands such as the 100DOH model [199]. Specifically, the baseline methods are the following:

Faster-RCNN [190] is a two-stage anchor-based 2D object detector on a single RGB image. In its classification head, the state change object is the only positive category. We train Faster-RCNN on our benchmark and use it to directly detect the bounding boxes of state change objects in PNR frames.

CenterNet [241] is another object detection method on a single RGB image. It estimates object keypoints to find object center points and regresses all other object properties, such as size, 3D location, and orientation. We train CenterNet to directly detect the bounding boxes of state change objects.

DETR [27] is an object detection model on a single RGB image based on Transformer [216]. It views object detection as a direct set prediction problem and uses a transformer encoder-decoder architecture to produce a set of object predictions including bounding box information as well as other information such as category. We train DETR to directly detect the bounding boxes of state change objects.

100DOH Model [199] first detects the bounding boxes of the human hand and objects as well as the relational vectors that links from each hand bounding box center to an object bounding box center. The final prediction of the objects are decided as the object predictions that satisfies the both the predictions of hand and relational vectors. We used the 100DOH model pre-trained on 100DOH dataset [199] to first detect hand bounding boxes and then predict state change object bounding boxes given the hands.

We show the number of state change objects and hand bounding boxes contained in our dataset in Table 17. The results of single-frame State Change Object Detection are illustrated in Table 18. All baselines struggle in detecting the State Change Objects with only one frame as input as an AP of 8-14%. There are several challenges in this task. First, the bounding box sizes of state change objects have large variance. For example, the size of state change objects can be as large as half of image in the action of "painting the wall"

Table 18. Results of single-frame State Change Object Detection. The performance is measured in Average Precision (AP).

Baseline	Backbone	AP	AP50	AP75
Faster-RCNN [190]	ResNet-101 [95]	13.4	25.6	12.5
DETR [27]	ResNet-50 [95]	15.5	32.8	13.0
CenterNet [241]	DLA-34 [233]	6.4	11.7	6.1
100DOH Model [199]	ResNet-101 [95]	10.7	20.6	10.1

and as small as a few pixels in the action of "igniting the match." Second, when only using one frame as input, the detection models did not consider the change of object appearance across different frames. As future work, we hope the researchers will investigate using models that take multiple frames as input and perhaps develop frameworks that incorporate tracking or association.

G.9 Discussion

This novel benchmark explores three aspects of objects undergoing state changes as a result of hand manipulation: the *when* (i.e. temporal localization of state change), *where* (i.e., spatial localization of objects that undergo change) and *what* (i.e., semantic notion of action and object transformation). As a first step, we have explored these independently using readily available localization and classification methods. However, approaches that aim to tackle this challenge should focus on jointly understanding the manipulation with its spatio-temporal impact on objects as these are transformed. For example, knowing an object is being split should offer a strong prior to the PNR localisation and detect two or more bounding boxes after the point-of-no-return. Such methods that tackle the dependencies between the tasks are yet to be developed. We hope this benchmark will spur innovative approaches that bridge the gap between action perception and the impact of actions on objects and environments.

G.10 Contributions statement

Kris Kitani helped formulate and write the object state change benchmark, designed the annotations and tasks for the HO benchmark. Dima Damen helped with the formulation and writing of the object state change benchmark, designed the annotations for the Hands and Objects (HO), and Forecasting benchmarks. Ilija Radosavovic coordinated HO data annotation, annotation analysis, and contributed to the definition and writing of the HO benchmarks. Rohit Girdhar helped coordinate the HO data annotation and annotation analysis. Abrham Gebreselasie adapted the SlowFast+Perceiver model for PNR temporal localization. Qichen Fu implemented all of the state change object detection baselines. Raghava Modhugu implemented the BMN baseline for PNR temporal localization. Kristen Grauman contributed to the formulation and writing of object state change benchmark. Siddhant Bansal helped with the processing of HO data, development of HO data loader for PNR temporal localization and implemented the I3D ResNet-50 baselines. Xingyu Liu was the lead coordinator and mentor of the HO benchmark baseline implementations, and also contributed to the definition and writing of HO benchmarks. Xuhua Huang developed of the initial SlowFast+Perceiver model. Yifei Huang implemented the Bi-directional LSTM baseline for the PNR temporal localization and state change classification.

H. Audio-Visual Diarization Benchmark

This section details the Audio-Visual Diarization (AVD) benchmark task definitions, annotations, baseline models, and results. As noted in Appendix B, the AVD benchmark uses only video where informed consent for capturing identities is explicitly collected from all participants in the scene, including faces and voice.

H.1 Motivation

Egocentric human perception is driven by inferring useful information from all the primary senses. While visuals captured by the eyes are one of the main *information channels*, sounds as captured by the ears are equally relevant. In particular, for understanding humans' interaction with the environment from the first-person perspective, detecting, localizing, tracking (both in 3D space and time) and understanding sounds by combining the necessary acoustic information with visual signals becomes even more critical. Several psychophysical studies have proven that humans are remarkably good at locating where a sound came from in 3D space with respect to their head position [156]. Sensitivity of humans to moving sounds in horizontal and vertical planes is also well documented [117, 178].

For a long time, the computer vision community has studyied the problem of precise localization of objects and people, robustly tracking and segmenting them using images. In this effort, we aim to bring audio (human speech in particular) into the mix. Truly audio-visual systems not only enable richer capture and analysis of the environment (and a user's interaction with it), but they also help build technologies for visually or acoustically impaired users (e.g., hearing aids, augmented reality).

The **goal of this benchmark** is to help advance the state of the art in audio-visual understanding from the egocentric viewpoint. Specifically, from a conversational perspective, the benchmark aims to understand *who is talking when, and about what*. From a visual perspective, we are also interested in *where the speaker is located*. Given an egocentric video, the proposed tasks require extracting the spatial location of the speakers, their voice activity across the length of the video, and the content of their speech.

Egocentric data presents several unique attributes to this problem. Firstly, sound sources may be visible within all, some, or none of the visual frames, depending on their movement within the scene and the movement of the camera wearer. Secondly, although the camera wearer is never visible (due the head mounted camera device) they are clearly audible and in fact often amplified compared to the other conversation participants due to the closeness to the microphone that captures the video. Third, natural dynamics in the scene (camera wearer walking, running, rapid changes in head movement etc.) add significant blur and distortion to the visual stream—some such noise is structured and relevant for understanding the context and semantic content in the scene.

H.2 Related Audio Visual Learning Work

There is a recent resurgence of work on audio-visual analysis within and beyond the computer vision community. These works tackle various aspects of audio-visual understanding, including source localization, cross-modal feature learning, audio spatialization, and audio source separation, as we briefly review next.

On audio-visual detection and tracking, recent works on multimodal learning explore ways to localize sounds in a given video frame [14,197,212] and infer spatialized sound from video [80,161]. Capturing and processing multi-channel audio is being studied in audio and microphone array signal processing communities, specifically from a user's perspective to understand a given scene [101, 169]. Building upon these, it is reasonable to expect that human-centric audio has information content that can directly improve visual object categorization and recognition. Indeed, this is observed in some recent work where audio disambiguates certain visually ambiguous actions [110, 226]. For actions and activity, audio events can also be directly used to perform summarization [13]. In particular, capturing ego-driven actions and activity and separating them from general background actions and activity in the scene is critical.

Alternatively, visual information has been used to disambiguate certain audio tasks like speech transcription. Specifically, audio-visual speech recognition has received a lot of attention in the last decade with multiple studies suggesting that automatic speech recognition (ASR) could benefit from visuals of the scene, or other non-acoustic information [5,104]. As shown in here, it is reasonable to expect that lip reading from a first person point of view would also benefit ASR systems.

In addition, audio-visual cross-modal learning may provide insight and solutions to one of the oldest problems in egocentric human communication ecology, referred to as cocktail party problem (CPP). The essence of CPP is "How do we recognize what one person is saying when others are speaking at the same time?" Human listeners must perceptually integrate the simultaneous sounds originating from one person's voice (e.g., harmonics and speech formants) and segregate these from the concurrent sounds of other talkers. In such situations, humans leverage visual information such as from lip movements to better understand, while their auditory system helps with focusing on a particular speaker characteristic while ignoring other speech/noise. Recent work on audio-visual diarization [83] and multimodal source separation from video show that CPP and its variations can benefit from visual signals [6, 57, 79, 81, 82, 171, 238].

Furthermore, humans are pretty good in understanding the context of a conversation even when words are incomprehensible. They are able to fill in the missing details using their context knowledge. This can be extended to sound sources that are non-humans as well. For a more detailed account of CPP please refer to [17]. Fully addressing CPP requires not only identifying and separating the different sound sources in the scene, but also understanding the auditory *attention* of the camera wearer—in other words, which sound source is the user attending to at the moment, or which one may the user want to attend to in the near future.

H.3 Related Datasets and Benchmarks

EPIC-Kitchens: [42,44] EPIC-Kitchens is among the most widely known ego-centric dataset with first-person view events and annotations. The dataset comprises of multi-faceted, audio-visual, non-scripted recordings in native environments, i.e. the wearers' homes, capturing all daily activities in the kitchen over multiple days. The dataset is 100 hours, 20M frames, 90K actions in 700 variable-length videos, capturing long-term unscripted activities in

45 environments using head-mounted cameras. Annotations are collected using a Pause-and-Talk narration interface. The dataset is widely used in action recognition, action detection, action anticipation, cross-modal retrieval, as well as unsupervised domain adaptation for action recognition.

VoxCeleb: [40, 165] VoxCeleb 1 and 2 comprise recordings of more than 6K speakers spanning a wide range of different ethnicities, accents, professions, and ages. The data is non-egocentric and is annotated for active speaker face bounding boxes, face tracks, and anonymous person IDs. VoxCeleb 2 in particular is defined for boosting research in speaker recognition, and it contains over a million utterances. Videos included in the dataset are shot in a large number of challenging visual and auditory environments. These include interviews from red carpets, outdoor stadiums and quiet indoor studios, speeches given to large audiences, excerpts from professionally shot multimedia, and even crude videos shot on hand-held devices. Audio segments present in the dataset are degraded with background chatter, laughter, overlapping speech and varying room acoustics.

VoxConverse: [39] VoxConverse is a related audio-visual diarization dataset consisting of over 50 hours of multi-speaker clips of human speech, extracted from YouTube videos. Similar to VoxCeleb, this data is also non-egocentric. This dataset was proposed to boost research in speaker diarization for audio-visual inputs. A bulk of the data instances are from political debates and news anchors so as to capture conversational scenarios with overlapping and interrupting speech.

AVA: [31, 192] The AVA spoken activity datasets are AVA speech and AVA active speaker. AVA speech is a densely annotated audio-based speech activity collection of AVA 1.0 third-person videos, and explicitly labels 3 background noise conditions, resulting in approximately $46,000$ labeled segments spanning 45 hours of data. AVA active speaker associates speaking activity with a visible face, resulting in 3.65 million frames labeled across approximately $39,000$ face tracks.

AVDIAR: [84] The closest egocentric dataset for audio-visual diarization is AVDIAR. It consists of 23 staged sequences, with each sequence duration ranging from ten seconds to three minutes (a total of 27 minutes of video). Each sequence comprises of 1-4 speakers some standing and some walking around in the visual FOV and having a conversation. The capture is done via a head mounted capture on a dummy head.

EASYCOM: [53] EASYCOM is a recent dataset open sourced for the purpose of boosting egocentric audio-visual learning research with a focus on multi-channel data and CPP. The dataset corresponds to 5 hours of conversational content with $3-5$ participants in a closed room setting. The content involves playing games, ordering food from a menu, and a general discussion on a prespecified list of topics. During the recording of the conversations, restaurant-like noise was played on loudspeakers in the room to mimic a real restaurant scene. The EASYCOM capture device use glasses with 6 mics attached to the frame. Although rich in terms of multi-channel egocentric acoustic content, the setup is constrained in terms of realism, the data is not in the wild, and most importantly the dataset is small.

Existing audio-visual datasets vs. Ego4D: Of these existing datasets, EPIC-Kitchens, AVDIAR and EASYCOM are egocentric. However, EPIC-Kitchens focuses on solitary activity by the camera wearer, and neither the video nor annotations accommodate audio-visual conversation tasks requiring multiple people. Although EASYCOM contains audio-visual conversation, it is a small dataset containing partly scripted conversations that are not in-the-wild. The participants in the sessions also do not move around. AVDIAR does include some participants who move around, but the camera wearer is a dummy head and, similar to EASYCOM, the data is not in-the-wild (sessions all are done in the same environment/scene). Ego4D accounts for all these aspects. Lastly, in contrast to VoxCeleb, VoxConverse and AVA, Ego4D offers first-person video and its conversation videos take place in casual daily-life environments with multiple speakers.

H.4 Tasks: Definition and Annotations

Here we detail the task definitions, the corresponding annotations, and the evaluation metrics. We propose a suite of tasks for the Audio-Visual Diarization (AVD) benchmark. These tasks are abbreviated as: *Localization & Tracking*, *Active Speaker Detection*, *Diarization* and *Transcription*. These tasks jointly capture who is talking when, to whom, and about what in a given egocentric conversational scene. Observe that these tasks are implicitly tied to each other; each subsequent task is driven in some form by a previous task (as further clarified in the task descriptions below).[16]

Task 1: Localization & Tracking: Where is the person in the visual field of view? This first task in AVD captures the spatial position of all the probable speakers in the scene, from the point of view of the camera wearer. The goal of the task is to compute bounding boxes for them. Unlike classical face detection benchmarks, this task is challenging in the sense that the dynamics of the camera wearer's head (coming from natural conversations) leads to significant movement in a speaker's apparent spatial location.

Annotations: For each speaker present in the 5 min clip a bounding box is provided. Each frame of the video is annotated for the task. We first utilized a face detection and tracking model to estimate these bounding boxes, and then a team of human annotators validated and corrected these machine-generated boxes to improve annotation quality. A bounding box is considered a valid human annotation if it captures 80% of the speaker's face; we peform a quality check steup to ensure this. Sideways looking faces are also annotated. Note that speakers who are very far from the camera wearer (oftentimes several meters away in the scene) and who do not come into conversational contact with the wearer are not annotated.

Evaluation: Recall that the goal of the task is to localize *as well as* track the speakers in the scene. Hence the evaluation metrics proposed account for the accuracy of trajectory of detected bounding boxes. We follow the standard multiple object tracking (MOT) metrics to quantify the speaker tracking results. There are many different MOT metrics, in which we are most interested in the MOTA in the CLEARMOT metrics [19], and IDF1, IDP, IDR in the Identity metrics [18]. MOTA, the multiple obtect tracking

[16] Note that although speech transcription and source localization are distinct from audio-only speaker diarization— all of which are well defined research paradigms in mainstream audio, speech and vision community— we cumulatively refer to all these together under the umbrella of audio-visual diarization for Ego4D.

accuracy, is a combined metric of false alarms, false positives and identity switches. MOTA is based on matching the tracking results with the ground truth at frame level, while the IDP (ID precision), IDR (ID Recall) and IDF1 (ID F1 score) are based on the tracking result to ground truth matching at the trajectory level. ID metrics give a tracker's performance on maintaining correct identification for each target.

Task 2: Active Speaker Detection: Who is speaking? The next task in AVD is to detect the active speaker in the scene. This task is in principle similar to active speaker detection—where the goal is to detect which of the visible people in the scene are speaking at a given time [192]. It builds on top of the previous localization and tracking task to recognize each of the speakers whose face bounding boxes are detected. Hence, this task does not take into account speakers who are not visible in the camera's FOV. Note that active speaker detection is also an important aspect of speaker diarization (which is the next task in the benchmark).

Annotations: We provide an anonymous speaker label (e.g., speaker 1, 2 etc.) for each speaker visible in the clip. The camera wearer is assigned the label C. This is done by utilizing the face bounding box tracks annotations and labeling each track one at a time. Hence, each face track gets assigned one unique label, and multiple tracks within a single clip may share the same label (corresponding to the same speaker). However, the labels are clip-specific, i.e., a speaker who may be present across multiple clips does not get assigned a shared unique label across the clips. Again, speakers who are never in the visual FoV are not assigned a label.

Evaluation: We use the object detection mAP to quantify the active speaker detection result. This is a frame-wise metric. In a video frame, if the intersection over union (IoU) between a detected face bounding box and the ground truth face bounding box exceeds a predefined threshold, i.e. 0.5, we have a positive face detection. Each detection has an associated class to indicate whether it corresponds to an active speaker. Active speaker detection methods give a confidence score of the active speaker class for each detected face bounding box [211].

Camera Wearer's Voice Activity Detection: Note that the camera wearer's face is never visible in the camera's field of view, and so they do not have any face tracks associated with them. However, in many cases, they are the dominant speakers. This is mainly because they are driving the interactions in many cases, and since their mouths are the closest to the microphones, their voice is in general amplified in the audio stream compared to other speakers. We propose to also consider them as active speakers and detect their voice. We use the object classification mAP to quantify the result of the camera wearer's voice activity detection.

Task 3: Diarization: Who spoke when? This next task further expands on the temporal aspect of active speaker detection (from the previous task). Given the set of speakers and their spatial localization in the visual field of view, this task aims to capture the voice activity of the speakers. It is identical to speaker diarization, a well studied research problem in the speech and audio domains [10, 177] and answers the question, "who spoke when". While speech from speakers that overlap with each other is one of the biggest issues to solve in this task, the egocentric perspective adds more complexity in terms of head motions and other dynamics associated with natural conversations. Note that the outputs of active speaker detection (the earlier task in the benchmark) also drive this task.

Annotations: For every active speaker label (where the annotations are from the previous Active Speaker Detection task), a human annotator marks the start and end time of that person speaking. We account for overlapping speech segments where multiple speakers talk over each other, but we ignore speech not relevant to the conversation such as background speech from a TV or speech further away from the camera wearer. Note that speech segments from the camera wearer are also annotated. The annotators rely both on the audio and the visual stream for creating these labels.

Evaluation: Diarization error rate (DER) is the *de facto* evaluation metric for speaker diarization [11], and it is well studied in the audio and speech processing community. DER measures the fraction of total time (in a given clip) that is not attributed correctly to a speaker or to non-speech. It is defined as follows:

$$\text{DER } (\%) = (E_{miss} + E_{fa} + E_{spk}) \times 100, \qquad (21)$$

where E_{miss} denotes the fraction of time that has been predicted to be non-speech while that segment is attributed to a speaker in the reference. E_{fa} denotes the fraction of time that has been predicted to be associated with a speaker, but is actually labelled as non-speech in the reference, and E_{spk} denotes the fraction of time where speech is associated with the wrong speaker. All errors are computed as a fraction of the total amount of speech.

Task 4: Transcription: What did the speaker say? The final task of AVD is to transcribe the speech of each speaker, i.e., performing ASR. Similar to the diarization task, some of the challenges associated with the transcription task include overlapping speech and environmental noise. In addition, the camera wearer's head movement results in a significant change of the audio volume of the speech recorded from others.

Annotations: Since the clips contain multiple speakers with overlapping speech segments and with different volumes, the final transcriptions are obtained in multiple passes. In the first pass, initial human annotations based on voice segments are merged with automatic annotations for regions with low volume. In a subsequent pass, human annotators had to correct and assign segments of transcriptions to the corresponding voice activity segments per speaker while also annotating overlapping speech. Note that annotators had both the audio and video available for annotation and, besides spoken words, the occurrence of other artifacts such as unintelligible speech or incomplete words have also been annotated. The final transcription annotations for a clip consist of a sequence of segments labeled with begin time, end time, transcript and speaker ID within the clip. In evaluations, we applied ASR to these segments individually and computed the performance over all of these segments. Please note that the time segments associated with the transcripts are not the same as the ones used in diarization because we separately annotated the overlapping regions here to reduce transcription errors and account for speakers talking in low volume. This allows us to also distinguish voice activity from speech activity. In addition, the use of time-segmented transcriptions is also slightly different from standard ASR datasets in speech community which mainly have text and no timestamps.

Evaluation: We utilize the Word Error Rate (WER), a standard ASR metric, for evaluating this task [114]. First, the minimum edit

or Levenshtein distance is computed between the reference and hypothesized transcription. WER then measures the ratio of the number of word substitutions (S), deletions (D) and insertions (I), i.e. the total number of edits necessary to convert the hypothesized transcription into the reference relative to the total number of words (N_w) in the reference:

$$\text{WER } (\%) = \frac{S + D + I}{N_w} \times 100. \tag{22}$$

H.5 Data Statistics

From across the 3,670 hours of video in Ego4D, approximately 764 hours of data contains conversational content, and are directly relevant for the AVD and Social benchmarks. Please refer to Section I.5 for a complete description of the experimental design and scenarios used in these sessions. From this set, a randomly chosen subset of 572 clips (each 5 minutes long) are annotated for this first version release. Of these 572 clips, 389 clips are marked for training, 50 clips for validation, and the remainder is the testing set.

Table 19 and Figure 43 summarize statistics about the speaker content from across these clips. Observe the long tails of mean and maximum number of speakers in the dataset. We note that in the first version of the data release, due to the fact that the total number of clips is relatively small, the test and/or validation batches may be biased in terms of changes in speakers' accents, changes in vocabulary usage (since the participants are from different cultural backgrounds from across the world), and in general changes in nature of interactiveness between speakers in a scene. There is marginal distributional shift among the training, testing and validation splits. This is mainly because of the smaller number of annotations in this version of AVD for Ego4D. We expect these distributional shifts to be less significant in future releases and as more data will be annotated.

Statistic (Avg.)	Value
Speakers per clip	4.71
Speakers per frame	0.74
Speaking time in clip	219.81 sec
Speaking time per person in clip	43.29 sec
Camera wearer speaking time	77.64 sec

Table 19. AVD Data Statistics.

H.6 Baseline Modeling Framework

Recall that the 4-part tasks in this benchmark are tied to each other, in the sense that representations learned from one task may be relevant for the others. To that end, we propose a baseline learning framework that addresses each task in a sequential fashion. The framework includes the following steps:

- We first detect people's heads and do short term tracking in the video. The short term tracker follows each detected head by expanding a set of trajectories based on their positions, sizes and the appearance of the person. The trajectories may end when occlusion happens or when the tracked person goes out of the field of view. New trajectories can also be added to the trajectory set.
- The short term tracker's trajectory for each person is often fragmented into multiple parts. Hence, we then optimize the grouping of the tracklets in step one so that the trajectories of each person can be linked together. We formulate the problem as a constrained combinatorial optimization problem. Integer programming can be used to solve the problem directly but it has exponential complexity. For efficiency, we develop a greedy approach which is much faster and still gives strong results.
- We then classify each person/head in each video frame as an active speaker or not. Based on the classification result and the corresponding detected long-term trajectories, we further associate the audio/speech to each person in the video. We use this preliminary list of audio feature embeddings to further extract and match un-associated audio segments to speaker labels.
- We then use two methods to detect the camera wearer's voice activity. The first method uses high energy audio segment in the clip (under the assumption that their voice has natural amplification compared to the remaining speakers). The second method is a deep classifier that predicts whether the wearer is speaking.
- Lastly, we applied ASR to the speech regions based on the ground truth segmentation and evaluated the WER across all segments. Evaluating the system by using another segmentation method is challenging especially in the case of overlapping speech segments. Jointly modeling time segments and transcriptions will be a challenging problem (as we discuss in Section H.7).

We describe further details about each of these steps below, and Tables 20–29 summarize the resulting performance metrics for the tasks.

Audio Only Models for Speaker Diarization The problem of speaker diarization from audio has been studied to a considerable extent in the field of speech processing [10, 177]. For the audio-only baseline system, the VBx diarization approach has been utilized [128] for having shown superior results on different types of datasets such as CALLHOME [3] (telephone conversations), AMI [28] (meetings) and DIHARD II [59] (myriad of domains ranging from audiobooks to YouTube videos). This method requires speech activity regions and these were obtained using the ASpIRE model based on a time delay neural network (TDNN) with statistics pooling, available with the Kaldi speech recognition toolkit [181]. We refer to this as kVAD (the Kaldi VAD model). Although this kVAD has been trained on slightly different data (telephone conversations), and thus does not provide the best possible results, it has been chosen for the baseline system because of its general availability.

The speech activity regions are uniformly segmented to obtain shorter segments and speaker embeddings (so-called x-vectors [206]) are extracted one per subsegment. The x-vectors are obtained with a ResNet101 extractor [96] trained to produce speaker-discriminative embeddings. The input to the network are log Mel-filter bank features every 10 ms, and given a segment of speech, it computes a single 256 dimensional vector that represents the

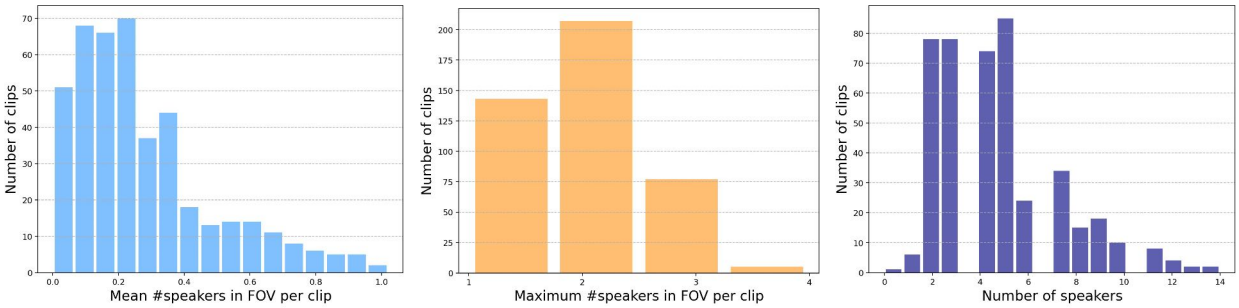


Figure 43. AV Diarization data statistics. Mean and maximum number speakers in FOV, and number speakers per clip.

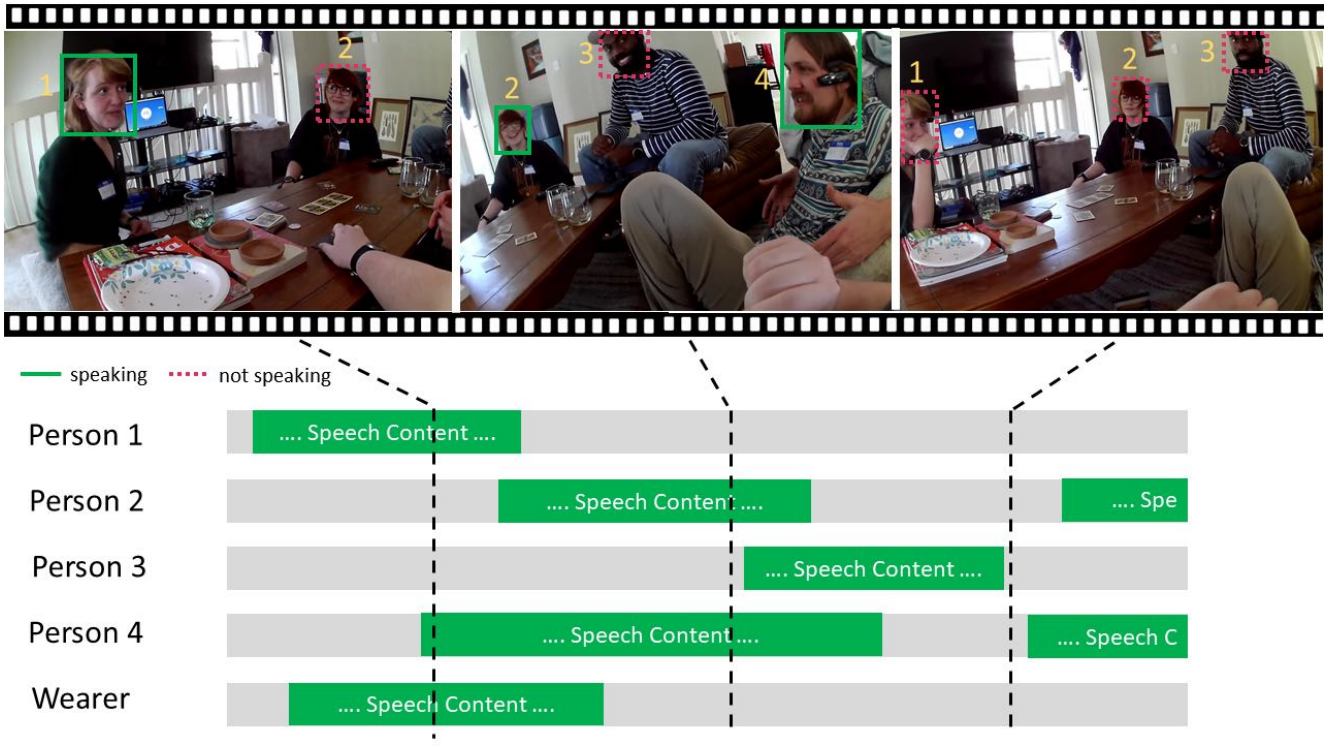


Figure 44. AV Diarization benchmark annotations summary. The four tasks are annotated in a sequential fashion, starting with localization and tracking of speakers, active speaker detection labels, diarization time stamps, and finally transcriptions. The figure shows the face detections (highlighted by bounding boxes), speaker detection (shown by the anonymous person IDs 1, 2, etc.), active speaker (highlighted in green) and voice activity (shown below in green highlighted time segments). Speakers in the visual FOV who are not talking are highlighted in dotted red boxes. The clips used for AVD (and Social Interaction) have consent from participants to leave their faces unblurred.

whole segment. The information of the whole segment is aggregated with a statistical pooling layer which computes the mean and standard deviation of activations over the time domain. A linear transformation is then used to reduce the dimensionality to 256. The training data consisted of VoxCeleb1 [165], VoxCeleb2 [40] and CN-CELEB [64] together, totalling 2877 hours of speech from 8178 speakers.

The x-vectors are initially clustered to a few dozens of classes using agglomerative hierarchical clustering. This initial clustering is fed as initialization to a Bayesian hidden Markov model which estimates altogether the number of speakers in the recording as well as the assignment of x-vectors to the states. Each state in the model corresponds to one speaker and the probability of observing a particular x-vector in a particular state can be interpreted as the corresponding speaker producing the corresponding segment of speech. The most relevant hyperparameters of the model were fine-tuned to obtain the best DER performance on the Ego4D validation set. The VBx implementation published by Brno University of Technology is publicly available as well as the training recipe published by Phonexia Research.

Short-term People Tracking The goal here is to track people's faces. However, our method can also be used to track the

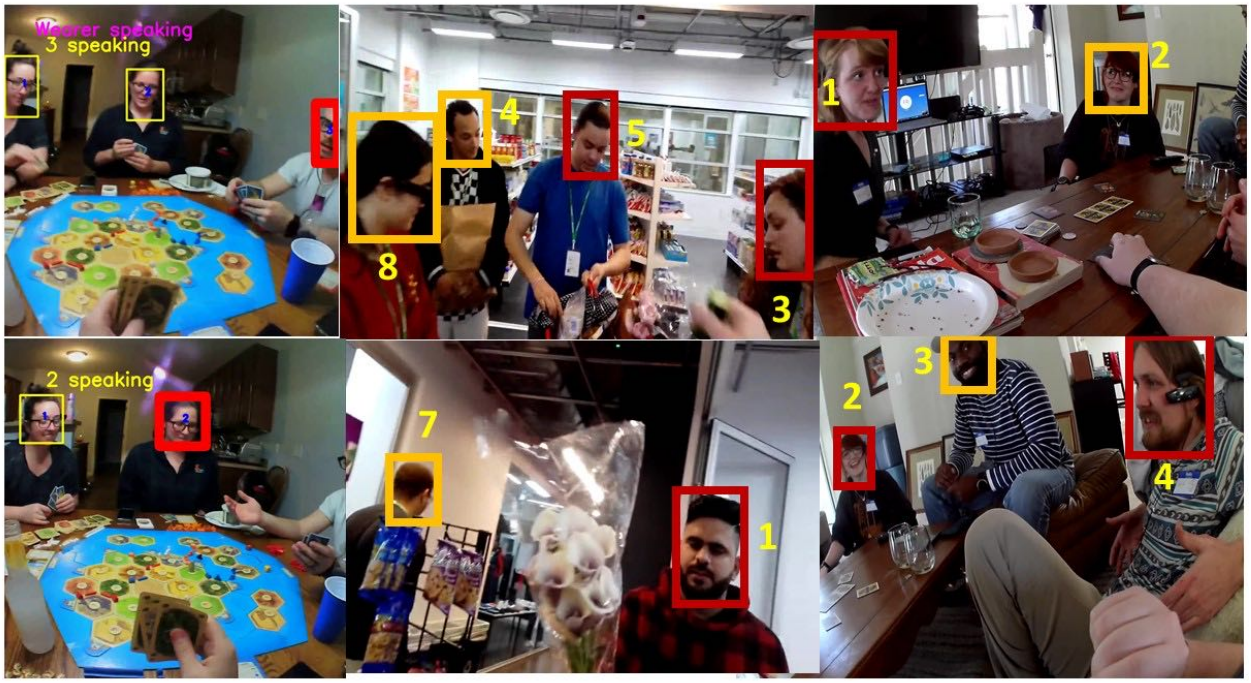


Figure 45. Example annotations showing the face detections (highlighted by bounding boxes), speaker detection (shown by the anonymous person IDs 1, 2, etc.), active speaker (highlighted in red) and voice activity (shown below in blue highlighted time segments). As illustrated here, the data for AVD includes people walking around and talking, sitting and playing games etc. The clips used for AVD have consent from participants to leave their faces unblurred.

whole body of each person. The short-term tracker maintains a set of trajectories. The trajectories include the attributes such as the person-ID, the frames tracked, a life counter, the appearance features and the positions of the tracked bounding boxes. Throughout, we use the term "person-ID" to refer to an anonmyous tag for a person in the video (person 1, person 2, etc.); no actual identities are available in the data, and the benchmark does not aim to perform any person identification. There are two kinds of trajectories. If a trajectory's tracked frames are less than a threshold, e.g. 5, it is in probation and is not counted as a real trajectory even though we maintain all the information for them. When a trajectory's tracked frames are greater than the threshold, it becomes a real trajectory. Each trajectory also has a life span. The life of a new trajectory starts from a fixed value. The life of a trajectory is restored to a fixed maximum value, such as 10, if the trajectory is matched to a candidate person head bounding boxes. Otherwise, the trajectory goes into a maintenance mode and its life decreases by 1 each time it fails to find a match. If the life of a trajectory goes to 0, it is removed from the trajectory set.

The key component of the short-term tracker is matching trajectories to the candidate head bounding boxes in each frame. This can be formulated as the following optimization problem:

$$\begin{aligned} &\min \sum_{(i,j)} c_{i,j} x_{i,j} \qquad (23)\\ &\text{s.t. } x_{i,j} \text{ forms a max-matching,}\\ &x_{i,j} = 0, \text{ if } (i,j) \in E,\\ &x_{i,j} = 0, 1, \end{aligned}$$

where $x_{i,j}$ is 1 if trajectory i matches candidate head box j and 0 otherwise. E is a set in which the pairs of trajectory and candidate cannot match each other, examples include cases such as the candidate is too far away, the size is too different or the appearance does not match. $c_{i,j}$ is the cost of matching trajectory i and candidate head detection j. This cost of matching, $c_{i,j}$, is computed as a linear combination of the normalized bounding box distances and the difference of the appearance features. The normalized bounding box distance is defined as the ratio of the Euclidean distance between the two corners of the last bounding box in the trajectory and the detected head bounding box in the image to the size of the detected bounding box. Each trajectory also maintains a feature vector to characterize the most recent appearance of the tracked person. This feature vector is obtained from a feature embedding network trained on a large person head dataset.

This optimization problem can be solved efficiently using the Hungarian algorithm or the primal dual algorithm. Due to the imperfect features, the optimization may have an identity switching

problem if two targets cross paths. To solve the problem, we enforce the longer trajectories to have higher priority to match. We use a two-step matching scheme. We first match all the trajectories that are longer than a specific threshold chosen empirically. Once done, we then match the shorter trajectories. This scheme naturally gives higher priority to longer trajectories, thereby reducing mismatches among them. This is more robust than a single stage matching where all trajectories are handled together.

In our implementation, the person detector is a Yolo-V3 detector [187] which detects the head and person bounding box simultaneously. The detector is trained on images from the Google OpenImage dataset [123] and a fisheye image dataset [73]. We use the detected head bounding boxes for people tracking. The person head appearance's feature is extracted using the person embedding network, which is trained on the VoxCeleb2 dataset using the triplet loss. The network has the structure of a ResNet-18.

Long-term Tracking by Trajectory Matching The short term tracker generates fragmented person trajectories. If a person is occluded or goes out of the field of view and reappears, it will receive a new ID. The fragmented trajectories are referred to as tracklets. We need to group the tracklets throughout the whole video to generate the final trajectories for each person. The grouping problem can be formulated as follows:

$$\begin{aligned} &\min \sum_{m,n} D_{m,n} y_{m,n} \qquad (24)\\ \text{s.t. } &y_{m,n} = y_{n,m}, \forall m, n,\\ &y_{m,k} + y_{k,n} \leq 1 + y_{m,n}, \forall m, n,\\ &y_{m,n} = 0, \text{if m and n overlap in time or } D_{m,n} > g,\\ &y_{m,n} \text{ is binary ,} \end{aligned}$$

where $y_{m,n} = 1$ if tracklet m and n can be grouped together and otherwise $y_{m,n} = 0$. $D_{m,n}$ is the appearance distance between the trackelet m and n and g is a threshold. Here $D_{m,n} = \min_{\{i \in T_m, j \in T_n\}} ||f_i - f_j||^2$, where T_i is the set of person head boxes in tracklet i and f_i is the corresponding feature embedding. The constraints require the grouping to be reflective: if tracklet m and n can be grouped together so can n and m, transitive: if m and k can be grouped together and so can k and n, then m and n can be grouped together. Two tracklets cannot be grouped together if they have time overlap or their distance is greater than a threshold g. The optimization can be solved using integer programming. However, this method has exponential complexity. We propose a fast greedy algorithm to solve the problem.

The greedy algorithm starts by treating each initial tracklet as a trajectory and progressively groups two trajectories with the closest D until no trajectories can be grouped together. Since the distance between two trajectories can be computed by finding the minimum of all the "element" tracklet pair distances, the merging procedure is efficient if we pre-compute and cache the element pair distance. This greedy approach gives strong results while maintaining low complexity.

The algorithm reduces to the minimum spanning tree method if there is conflict between each pair of trajectories. However, if there are time-conflicting tracklets, there is no guarantee the greedy algorithm gives the globally optimal solution. We illustrate the method through a simple example: Assume there are trackelets $\{T_1,T_2,T_3,T_4\}$, T_1 and T_2 have time conflict, and T_3 and T_4 have time conflict. D(T_1,T_3) = 10, D(T_2,T_4) = 1, D(T_1,T_4) = 3 and D(T_2,T_3) = 4. We assume $g = 20$. Using the proposed greedy method, the solution P is $\{\{T_2,T_4\},\{T_1,T_3\}\}$ whose overall cost is 11. However, the optimal solution is $\{\{T_1,T_4\},\{T_2,T_3\}\}$ whose overall cost is 7. Even though the greedy method does not guarantee the global optimal solution, empirically we observe that the proposed method give strong results. In fact, if the person embedding is accurate, these corner cases would probably never occur and the greedy result would approach the globally optimal solution.

Metric	Valid	Test
MOTA	74.52	71.94
MOTP	79.07	79.17
IDF1	84.92	80.07
IDR	80.40	73.52
IDP	89.97	87.90

Table 20. Localization and tracking baseline metrics on the validation and the test sets respectively.

Table 20 summarizes the tracking metrics MOTA, MOTP, IDF1, IDR, and IDP on the validation and test sets.

Active Speaker Detection: We use two approaches for active speaker detection. One approach is based on mouth region classification, and the second method is a transformer based audio-visual method for active speaker detection [211].

RegionCls: Our first approach is based on the classification of mouth regions. It first computes the 3D head orientation using a regression network. In our implementation, the z direction is into the image; if the head 3D orientation z coordinate on the unit sphere is greater than 0.3, we assume the face is away from the camera. If the face is facing away from the camera, we ignore the image and the active speaker detection result is set to null. For faces looking at the camera, our method first regresses the facial key points using the image within the person's head bounding box. We use the mouth key points to crop out the mouth image. The cropped mouth image is then sent to a classification network to classify whether the speaker is talking or not.

Note that we also explored using multiple images, wherein we stack a short sequence of cropped mouth images in a time interval for active speaker classification. Our experiments show the multiple mouth images input do not significantly improve the result. This is probably due to the fast movement of the camera and sometimes difficult angles of the face. This causes inaccurate cropped mouth regions.

TalkNet: [211] TalkNet is an end-to-end pipeline that takes the cropped face video and corresponding audio as input, and decides if the person is speaking in each video frame. It consists of a feature representation frontend and a speaker detection backend classifier, as illustrated in Figure 46. The frontend contains an audio temporal encoder and a video temporal encoder. They encode the frame-based input audio and video signals into the time sequence of audio and video embeddings, representing temporal context. The backend classifier consists of an inter-modality cross-attention mechanism to dynamically align audio and visual content, and a

Algorithm 1 Greedy Tracklet Grouping

Initialize sets P={S_1,S_2, ..., S_N}, where $S_i = \{T_i\}$, T_i is the tracklet i and N is the number of tracklets.
for (m, n), m=1..N and n=1..N **do**
  compute $D(m, n)$
end for
while True **do**
  for (S_m, S_n), $S_m \in P$ and $S_n \in P$, and (S_m, S_n) do not have time conflict **do**
    compute $F(S_m, S_n) = \min_{T_a \in S_n, T_b \in S_m} D(a, b)$
  end for
  (m^*, n^*) = argmin($F(S_m, S_n)$)
  if (m^*, n^*) is empty or $F(S_{m^*}, S_{n^*}) > g$ **then** break
  end if
  $S_{m^*} = S_{m^*} \cup S_{n^*}$ and P.pop(S_{n^*})
end while
P includes the grounded trajectories

self-attention mechanism to observe speaking activities from the temporal context at the utterance level.

Tables 21, 22, 23 and 24 summarize the resulting performance. For each of the two proposed baseline models, we report performance summaries with pretraining based on AVA and also models trained using only videos from the Ego4D training dataset. Note that the video-only approach can be combined with any voice activity detection to remove false alarms. Here we use such an algorithm from [203], and we refer to this as sVAD This can greatly improve the active speaker detection results. The max-filtering has a window size of 11. TalkNet also has a built-in smoothness filtering to post-process the raw classification result.

Model	mAP@0.5
RegCls w/o smoothing	29.68
RegCls + max-filtering	31.95
RegCls + max-filtering + sVAD	33.72
TalkNet	34.75
TalkNet + sVAD	34.56
Always Speak	24.46

Table 21. Active speaker detection baseline metrics on the test set with pre-training using AVA. In Always Speak, all the detected faces are classified as active speakers.

Model	mAP@0.5
RegCls w/o smoothing	29.65
RegCls + max-filtering	32.77
RegCls + max-filtering + sVAD	34.35
TalkNet	50.90
TalkNet + sVAD	49.66

Table 22. Active speaker detection baseline metrics on the test set using training videos in the Ego4D dataset.

Model	mAP@0.5
RegCls w/o smoothing	22.09
RegCls + max-filtering	22.88
RegCls + max-filtering + sVAD	25.53
TalkNet	34.36
TalkNet + sVAD	34.65
Always Speak	20.94

Table 23. Active speaker detection baseline metrics on the validation set with models trained on AVA dataset. In Always Speak, all the detected faces are classified as active speakers.

Model	mAP@0.5
RegCls w/o smoothing	20.33
RegCls + max-filtering	21.93
RegCls + max-filtering + sVAD	24.60
TalkNet	51.04
TalkNet + sVAD	50.58

Table 24. Active speaker detection baseline metrics on the validation set using training videos in the Ego4D dataset.

Matching Speakers Outside FoV: Based on the tracked heads and the active speaker detection results, we can associate the audio to the visible people in the scene. However, this is still not complete because there are cases in which the speaker is outside of the visual field of view. To solve this problem, we first create an audio-signature for each visible person in the video.

We extract one second of audio centered at each video frame time instant. If the audio corresponds to a speaking head in the image, we compute the audio embedding of the one second audio and insert the feature into the audio signature library of the person. The audio embeddings can be obtained from any speech representation learning methods. We explored several models including a modified ResNet18 which takes audio spectrogram logarithm magnitude in one-second windows as the input and trained on the VoxCeleb2

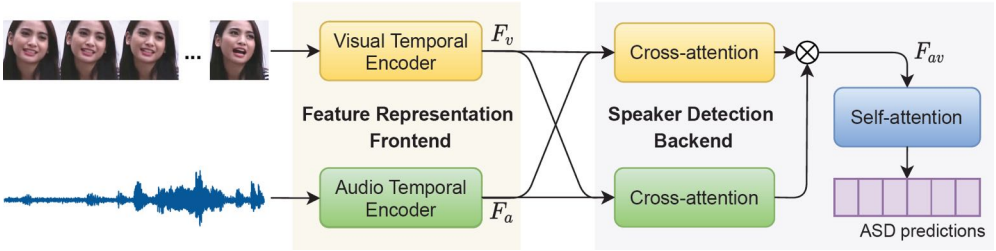


Figure 46. TalkNet: An audio-visual temporal network for detecting and tracking the active speaker in a video [211]. Figure is from [211].

dataset using triplet loss, and a version of wav2vec 2.0 [15]—a self-supervised approach to speech representation learning.

We parse the video and find instants when a particular person is not in the video frame and match the audio embedding to the person's audio signature library. We find the minimum distance of this audio embedding to all the signature audio embeddings in the library. If the distance is less than a predefined threshold, we classify the person as speaking and otherwise not. Note that the audio embedding is used only within the same 5 minute video clip and never across video clips. Person IDs are always anonymous tags (person 1, 2, etc.).

We use this method to detect all the background audio of the people of interest when they are not visible. This method assumes that the active speaker is perfect. In reality, active speaker gives noisy results. This would cause other people's voice feature to be included in a person's signature library and affect the final audio classification result.

Tracking Camera Wearer's Audio: The camera wearer is a special participant because their face is invisible in the egocentric videos. The active speaker detection method thus cannot be used to associate the wearer with their voice. We use two methods to detect the camera wearer's voice.

Method I: The first method uses energy filtering followed by audio matching. This method does not need ground truth labeling of the camera wearer's voice activities. Since the microphone of the camera is usually closer to the wearer's mouth than other subjects in the scene, the amplitude of the wearer's voice often has higher energy than other participant's voices. We use this heuristic to extract candidates of the wearer's voice by choosing portions of audio with energy higher than certain threshold. Since different recordings have different levels of loudness, we normalize the audio using the maximum energy and then choose the possible wearer's voice using a fixed percentage of the maximum energy. This threshold percentage is set to be as high as possible to avoid false alarms. Once the candidate audio is selected, we use the same audio matching method described in the previous section to find all the audio that belongs to the camera wearer. This simple method works reasonably well as summarized in Table 25. The approach fails when the wearer never talks or talks in a very low voice, and in general the baseline works better for near range microphones than long range microphones.

Method II: In the second method, we directly classify the audio at each time instant to two categories: wearer's voice or not wearer's voice. The logarithm magnitude of the spectrogram at 40ms window is the input. The network is a modified ResNet. The network is trained on the Ego4d AV training dataset using a standard cross-entropy loss.

We use classification mAP to quantify the wearer audio activity detection result. We report the average mAP on both the test videos and validation videos in Table 25.

Model	Valid	Test
Method I	43.95	50.61
Method II	72.00	74.29
Always Speak	21.30	26.09

Table 25. Camera wearer activity detection baseline metrics (mAP) on the validation and test sets respectively. Always Speak assigns that the wearer speaking in each video frame.

Speaker Diarization Tables 26 , 27 and 28 summarize the speaker diarization DER metrics for the baseline models proposed in the earlier sections. We report the results with training only on Ego4d data as well as on with training on existing diarization datasets. Note that the audio-only DER is aggregated while the audio-visual DER is averaged. Also note the impact of the VAD on the diarization performance with the audio-only baseline. It should be noted that a model more tailored to Ego4D-like data could be used to obtain better performance. Nevertheless, this aspect still poses challenges on the AVD benchmark.

Transcription To obtain baseline transcriptions, we used the pre-trained Gigaspeech model provided in the ESPNet model zoo [1]. This model is trained on the Gigaspeech dataset [34] which contains 10000 hours of speech. Input features to the model are logmel features augmented using the SpecAugment method [173] and normalized by global mean-variance normalization. The encoder of the acoustic model is based on macaron-style conformer [93] with 12 blocks and 8 attention heads and the decoder is based on a 6-layer transformer [217] with 8 attention heads. In both the encoder and decoder, linear layers have 2048 units and the encoder output is 512 dimensional. The decoder output has 5000 sentencepiece [122] units. The model is trained using a joint CTC and attention objective [112]. For decoding, no language model is used. For decoding, we used CTC weight of 0.3 and beam size 20 which we did not fine-tune on the Ego4D dataset. The pre-trained model obtained from [1] cannot support 5-min videos, hence, we used oracle segment information from the transcription annotations to segment the data and we decoded each segment separately. The

Model	trained on Ego4D	sVAD	DER [%]
RegionCls	no	no	84.79
RegionCls	no	yes	83.88
TalkNet	no	no	86.68
TalkNet	no	yes	85.85
RegionCls	yes, only	no	80.52
RegionCls	yes, only	yes	80.17
TalkNet	yes, only	no	73.14
TalkNet	yes, only	yes	73.32
Always Speak	-	-	>100
Never Talk	-	-	100

Table 26. Diarization Baseline Metrics showing DER on the test set. In Always Speak, all the detected people are labeled as "speaking" in each video frame. In Never Talk, all the detected people are labeled as "not speaking" in each video frame.

Model	trained on Ego4D	sVAD	DER [%]
RegionCls	no	no	98.82
RegionCls	no	yes	90.98
TalkNet	no	no	99.73
TalkNet	no	yes	92.14
RegionCls	yes, only	no	81.66
RegionCls	yes, only	yes	79.97
TalkNet	yes, only	no	80.58
TalkNet	yes, only	yes	79.30
Always Speak	-	-	>100
Never Talk	-	-	100

Table 27. Diarization baseline metrics showing DER on the validation set. In Always Speak, all the detected people are labeled as "speaking" in each video frame. In Never Talk, all the detected people are labeled as "not speaking" in each video frame.

Type of VAD	Valid	Test
kVAD	67.24	65.28
Ref. Activity	36.56	39.99

Table 28. Diarization performance with audio-only models for validation and test sets using kVAD and reference (ground truth) voice activity annotations.

final WER is obtained by counting the total number of errors over the whole validation or test set.

In Table 29, we summarize the WER results depending on the VAD segmentation method on both validation and test sets. To compute the final WER, we 1) removed punctuation from both the reference and the ASR hypothesis, 2) allowed soft-match on contractions such as (I will vs. I'll) using the English global mapping file from Kaldi repository [2], and 3) used the NIST sclite tool [72]. As we can see from Table 29, on both the test and validation sets, the WERs are quite high. This shows that the dataset is challenging for an off-the-shelf ASR model because of overlapping speech, noise, different volume levels for different speakers, occasional foreign word usage, etc.

Speech Segments	Valid	Test
Ground Truth	64.8	59.2

Table 29. ASR transcription WERs (%) on the validation and test data using the reference speech segmentation.

H.7 Discussion

Although AV diarization presents a task suite composed of reasonably well understood tasks from the vision, speech and audio communities, our baseline results clearly suggest that efficient speaker localization, tracking, diarization and transcription is a rather complex problem in the egocentric perspective and with in-the-wild data. This is specifically evident from the performance of the joint audio and video driven diarization and transcription baselines (with DER of $> 80\%$ and WER of $> 60\%$). Overlapping speech makes both these tasks particularly difficult to annotate as well as evaluate any proposed models. Performing some audio-visual source separation prior to these tasks may improve the efficacy, nevertheless sensitivity to changes and difference in speech amplitudes of overlapping speakers would still be challenging to address.

Novel cross-modal learning approaches that jointly model audio and visual modalities while accounting for such attributes (overlapping speakers, interruptions, noise in the wild etc.) are needed to further improve these performances. The baseline framework we utilized here also does not account for efficient information sharing across the four tasks in the benchmark. Specifically, the relationship between robust localization and tracking with multi-speaker diarization is not studied, and this is also not well understood in the literature. We expect this to be a challenging problem.

We also observed that subjective attributes in conversations, like speaker accents, changes in vocabulary usage based on cultural differences etc., influence both the content of the speech and the clarity with which it can be captured in human annotations. The camera wearer's head motion adds significant blur to speakers' faces. To account for such aspects we performed quality checks on human annotations, and we expect novel unsupervised and self-supervised learning will help further address such subjective attributes.

In future versions, we expect to increase the scope of the task suite (i.e., proposing new tasks and annotations), thereby opening new avenues for both core machine learning in first person perspective, and also for robust multi-modal representation learning. We could also investigate research directions focused on spatial audio by creating 3D environments coupled with SoundSpaces [32]. This enables new research and tasks in audio-visual sound source localization, audio-visual direction-of-arrival estimation and related immersive reality applications. We note that a small fraction of our dataset does comprise of binaural audio captured using in-ear microphones and an audio recorder (Tascam, Appendix A).

H.8 Contributions statement

Vamsi Krishna Ithapu co-led the audio-visual diarization benchmark workstream, the corresponding tasks definition, data selection methodology, data annotation tooling and guidelines and writing. Christian Fuegen co-lead the audio-visual benchmark workstream, the diarization and transcription tasks definition, the corresponding annotation guidelines and paper writing. Hao Jiang worked on data annotation tooling, tasks definition for localization and tracking, active speaker detection and diarization; also worked on building the baseline models for these tasks and writing. Federico Landini and Jachym Kolar worked on baseline models for audio-only voice activity detection and diarization, and writing. Leda Sari worked on transcription task definition, corresponding annotation guidelines and baseline modeling. Eric Zhongcong Xu worked on data selection methodology and the baseline modeling of active speaker detection. Ruijie Tao and Mike Zheng Shou worked on the modeling of active speaker detection. Hanbyul Joo worked on data annotation tooling and data selection methodology. Christoph Feichtenhofer worked on the task definition and metrics. Anurag Kumar worked on active speaker detection and diarization tasks definition, and on audio embeddings modeling for these tasks. Morrie Doulaty worked on baseline models for voice activity detection and diarization and data analysis of annotations. Lorenzo Torresani worked on the tasks definition and annotation guidelines. Kristen Grauman contributed to the benchmark formulation and writing.

I. Social Interaction Benchmark

This section details the Social Interaction benchmark task definitions, annotations, baseline models, and results. We also provide details on the video data collection process for multi-person capture with participants who consented to have their faces unblurred and conversation recorded (Appendix I.5). As noted in Appendix B, the social benchmark videos were screened to remove any information (e.g. last names or social media accounts) that could directly identify participants. However, participants' faces and voices are present as per our informed consent.

I.1 Formal Task Definition

LAM and TTM are defined as follows: (1) LAM: $y = f(\mathbf{I}, \mathbf{B})$; (2) TTM: $y = f(\mathbf{I}, \mathbf{A}, \mathbf{B})$ where $\mathbf{I} = \{I_t\}_{-T_1}^{T_2}$, $\mathbf{A} = \{A_t\}_{-T_1}^{T_2}$, and $\mathbf{B} = \{B_t\}_{-T_1}^{T_2}$ are time-synchronized past sequences of video, audio, and bounding boxes, respectively, where T_1 and T_2 are the length of the past and future time horizon, respectively, and $t = 0$ is the center frame. The bounding box indicates the target person to classify. y is a binary classification label defined by:

$$y = \begin{cases} 1 & \text{if target looks/talks at camera wearer} \\ 0 & \text{otherwise.} \end{cases} \quad (25)$$

The LAM and TTM tasks are defined as a frame-level prediction y, which stands in contrast to audio analysis tasks where labels are often assigned at the level of audio frames or segments. A desired model must be able to make a consolidated decision based on the video and audio cues over the time course of an utterance. For example, if the speaker turns their head to the side momentarily while speaking to the camera-wearer, then a frame where the speaker is looking away would have $y_{\text{LAM}} = 0$ while $y_{\text{TTM}} = 1$. Figure 47 gives some frame level visualization of annotations that illustrate the task definitions.

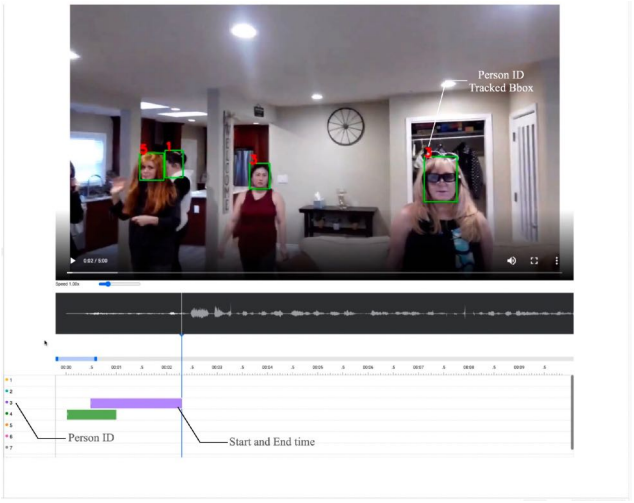
I.2 Annotation Statistics

The social task annotations, LAM and TTM, build on the same video clips used in the AV diarization tasks and described in Appendix H.5. Fig 48 summarizes the statistics of LAM and TTM annotations across these clips. We compute the percentage of the frames with LAM or TTM annotations in each clip and show the histograms in Fig 48 (a) and (b), respectively. In many clips, these events happen rarely (10 % or lower), and the frames with LAM annotations are less frequent than TTM cases. We also list the duration of each LAM or TTM annotation (the duration between start and end time) in Fig 48 (c) and (d), in order to illustrate the significant variations in length. The most frequent case is short-duration LAM or TTM behaviors, lasting 1 or 2 seconds. The data was organized as follows for baseline model training in Section I.3: 389 clips were held out for training, comprising 32.4 hours in total. An additional 50 clips (4.2 hours) and 133 clips (11.1 hours) were held out to form the validation and testing sets, respectively.

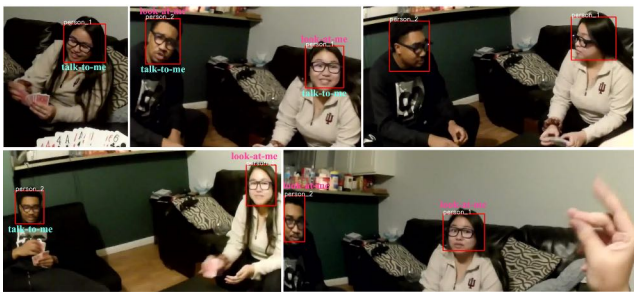
I.3 Social Baseline Models and Results

LAM Our baseline model for LAM is a video-based model using ResNet-18 and Bidirectional LSTM. Our model uses the cropped face regions in video as input in order to focus on cues about the head pose and social attention visible in the face. The architecture of our baseline is similar to the Gaze360 [111]. As illustrated in Fig 49(a), we input 7 consecutive frames ($T_1 = 3$ and $T_2 = 3$) from one face tracklet, and each image is resized to 224×224. Each frame is then processed by the ResNet-18 backbone independently to generate 256 dimensional face features. The feature sequence is encoded by a Bidirectional LSTM, which has two recurrent layers with dimensionality 256. The output is fed into a classification head to predict the binary LAM result for the center frame at the t-th timestamp. The LAM task has a class imbalance issue, and we use weighted cross-entropy loss. Since the architecture is similar to Gaze360, we have two options for the initialization: first, initializing the backbone from a pretrained Gaze360 model; second, initializing the model randomly and training from scratch on Ego4D. During training, we sample center frames with a stride of 3. The network is optimized by Adam with a learning rate of 5×10^{-4}.

The results are shown in Table 30. Our baseline model achieves an mAP of 66.07% on the test split when initialized randomly, and

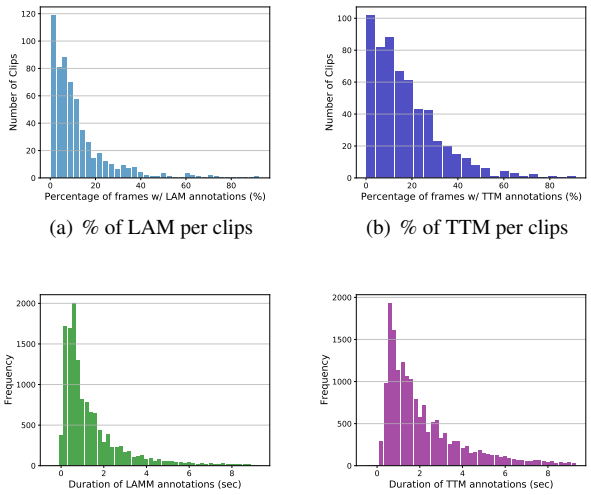


(a) Annotation tool



(b) Visualization of annotations.

Figure 47. (Top) The GUI of the annotation tool; (Bottom) Visualization of example annotations. Note that LAM (denoted by magenta text) and TTM (denoted by cyan text) may not necessarily occur together as shown in these examples.



(a) % of LAM per clips (b) % of TTM per clips

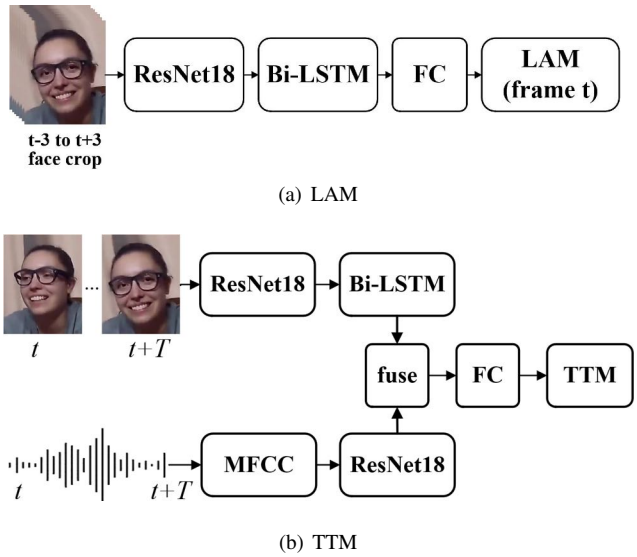
(c) Duration of LAM (d) Duration of TTM

Figure 48. Social task annotation statistics. (a) Histogram showing the number of clips vs. the percentage of frames with look-at-me annotations; (b) Histogram showing the number of clips vs. the percentage of frames with talk-to-me annotations in each clip; (c) Histogram showing the duration of look-at-me annotations; (d) Histogram showing the duration of talk-to-me annotations.

the performance is higher at 78.07% when initialized from Gaze360. These findings highlight the close relationship between the LAM task and gaze estimation. The random guess model achieves about 8% accuracy because the negative samples account for 92% of the test split and the model always predicts looking at me.

TTM The baseline model for TTM digests multi-modal inputs: each audio segment is paired with an associated face crop. Since the audio segments vary substantially in duration, we break the long utterances into short segments whose maximum duration is limited to 1.5s. If the segment is shorter than 0.15s, we skip it in the training stage. The associated faces are also resized to 224×224, and the video encoder is the same as LAM. However, sometimes the speakers leave the field of view or become invisible due to the rapid motion. In this case, we pad the face sequences with blank images. The MFCC feature is extracted every 10ms with a 25ms window length. The feature is then fed into the audio backbone, a ResNet-18 designed for audio tasks [38]. Following the encoders, we concatenate the audio and visual embeddings and pass them to the final classification head to get the TTM result for the visible faces associated with the segment. To train the model in parallel, we first sort the short segments based on the length and group the segments into a batch if they have the same duration. The batch size is restricted by the GPU memory; we use a batch size of 400. The model is also optimized using Adam with a learning rate of 5×10^{-4}.

Table 31 summarizes the TTM results. TTM is more challenging than LAM. We can see that our baseline model only increases the mAP by 9.77% on the test split in comparison to the random guess model.



(a) LAM

(b) TTM

Figure 49. **Baseline model architectures.** (a) LAM model uses a ResNet-18 as a backbone to extract the feature of each frame. A Bidirectional-LSTM then takes the sequence and encode the features into one embedding. We pass the embedding to FC layer that predicts the LAM result. (b) TTM model has two encoders. The video encoder is the same as LAM. The audio encoder extracts the MFCC frequency map of the audio segment and the feature is fed into a ResNet-18 network. The visual and audio embeddings are concatenated and passed through the FC layer to predict the target of this utterance.

I.4 Discussion

While the benchmark tasks of detecting when attention and speaking behaviors are directed towards the first-person are closely related to existing analysis tasks, it is clear from the baseline performance that there is substantial room for improvement, with mAP of 78.07 for LAM and 55.06 for TTM.

The TTM task is particularly challenging because it requires analysis of the audio content to understand the target audience of an utterance, as well as the fusion of audio and video cues. The most complete solution to this problem will require an understanding of the semantics of the utterance in the context of an evolving conversational interaction. Future work on this task might involve more sophisticated language modeling and possibly hierarchical analysis approaches that allow the integration of cues at multiple levels, e.g. at the dialog level to understand who is participating in a conversational exchange, at the utterance level to access semantics, and at the audio level to exploit prosodic and other cues.

The LAM task presents additional challenges such as the need to deal with motion blur and fast head movements, and may also benefit from a more explicit modeling of head movement and the patterns of gaze behavior that arise in conversational interaction.

I.5 Social Dataset Collection

The Ego4D Social data collection process was designed to achieve: 1) naturalistic interactions, 2) multi-modal capture, and 3) diverse

	val		test	
	Acc	mAP	Acc	mAP
Random Guess	8.57	51.19	7.98	50.96
Baseline (Gaze360)	91.78	79.90	87.97	78.07
Baseline (Random)	86.45	72.11	75.38	66.07

Table 30. **Results of LAM.** The baseline model was initialized from Gaze360 [111] (2nd row) and at random (3rd row).

	val		test	
	Acc	mAP	Acc	mAP
Random Guess	32.44	53.82	47.41	50.16
Baseline	64.31	56.50	49.75	55.06

Table 31. **Results of TTM.** The baseline model is initialized randomly.

participants and environments. Participants consisted of friends and family groups and data was captured in residences and local neighborhoods, ensuring naturalistic interactions. Capture hardware varied across sites but included wearable cameras, wearable eye trackers (at Georgia Tech and Indiana University), binaural recording systems, and smart watches (at Georgia Tech). Protocols included highly-structured settings, where participants were asked to play games over a period of a few hours in a residence, and unstructured settings where participants captured social interactions in daily life over a period a week or more. Sample social interaction contexts included playing board and card games, preparing meals, and going on walks. The bulk of the data collection took place during the COVID-19 pandemic, and the resulting study protocols were designed to safeguard participants against additional risk.

The social data consists of data collected at five sites: Atlanta, Bloomington, Redmond, Twin Cities, and Singapore. In total, 764 hours of video and audio were collected for the social benchmark task. A detailed summary of the data collection practices at each site can be found in Appendix A.

I.6 Derived Tasks for Future Social Benchmarks

The core tasks of LAM and TTM define a starting point for analyzing multi-modal egocentric data and inferring social interactions. We now describe two groups of potential future tasks, attention tasks and speaking tasks, that could be supported via the existing annotations in Ego4D Social and the gaze data collected from eye trackers by Indiana University and Georgia Tech.

Egocentric Attention Prediction (EAP) Prior work [135, 137] has demonstrated the feasibility of predicting where the camera-wearer is looking (i.e. their egocentric attention) using only egocentric video captured from a head-worn camera. This work leveraged the context of hand-eye coordination tasks, which require gaze to be coordinated with hand movements and objects. A subset of the Ego4D Social data includes gaze measurements produced by wearable eye trackers by Indiana University and Georgia Tech participants (e.g., Pupil Invisible), which will greatly expand the size of data for hand-eye coordination in the wild.

Social Gaze Prediction (SGP) The LAM task addresses the special case of social gaze: a person looks at the camera-wearer. It is possible to generalize the task by predicting the social gaze target for each of the visible faces in an egocentric video, i.e., $y^p \in \{0, 1, \ldots, M\}$, where M is the total number of participants in a group social interaction, and $p \in \{0, 1, \ldots, M\}$. p is the index for social members. The case $y^p = q$ means that target p was looking at participant q. The case $y^p = 0$ captures alternative gaze targets, including non-social gaze targets (e.g. looking at an object), looking at people who are not wearing an egocentric camera (with the result that ground truth annotations are not available), and looking at unknown targets not captured in any of the egocentric videos. Let $\hat{y}^{q,p}$ denote the LAM label for target person p visible in frame of egocentric video $\mathbf{I}_q$ captured by participant q. Then the SGP label is given by $y^p = \arg\max_q \{\hat{y}^{q,p}\}$. The Ego4D Social data includes synchronized videos from multiple social members, which will allow us to expand the annotation by matching the person ID with the camera-wearers. Note that since the video recorders are not genlocked, the identification of corresponding frames will only be approximate. However, since gaze behaviors persist over multiple frames we do not believe this will be an issue.

A key issue in defining the task is the determination of the participant set. For a 2D version of SGP, termed SCG-2D, the participant set is defined by participants who are visible in frame t. This is a social version of the video-based gaze follow task [37], where the goal is to predict whether each target participant is looking at any of the other participants who are visible in the frame. A more challenging 3D version of the task, SCG-3D, uses all of the participants who are present in the social scene at the time of frame t. This task requires the ability to predict which participant the target person p is looking at in the case where that participant is not visible in frame t. This can in principle be accomplished by maintaining a birds-eye view layout map of the social scene, that captures the approximate spatial relationships between the participants. Such a layout map could be used in conjunction with an approach like Gaze360 [111] to solve the SCG-3D task. Note that this task could potentially benefit from taking recorded binaural audio as an additional input, as the ability to localize sound sources could provide additional cues for determining the locations of gaze targets which are not visible in the video.

Utterance Target Prediction (UTP) The TTM task can be generalized to the full set of participants in the same way that LAM can be extended to SGP. The input space is the same as TTM and the output space is similar to SGP, where $y^p = q$ means that participant p is talking to participant q, and $y^p = 0$ denotes the cases where the participant is not talking to anyone, or is talking to someone who is not wearing an egocentric camera (and therefore ground truth cannot be determined). In contrast to SGP, UTP requires the identification of all of the target recipients of an utterance. In fact, our TTM annotation already supports this task, as it differentiates the case where the utterance is directed to multiple participants including the camera wearer. This additional label is ignored in the design of the simpler TTM task.

Transcript-based Variants For all of the previously-defined social tasks it is possible to define a variant which utilizes a transcript of the audio file as an additional input modality. For example, the TTM-T task is the variant of TTM with the prediction defined

as $y^p = f(\mathbf{I}, \mathbf{A}, \mathbf{T}, \mathbf{B})$, where $\mathbf{T}$ the transcript (time-stamped sequence of words) obtained from $\mathbf{A}$. This can potentially simplify the use of dialog cues to identify the intended targets for utterances and social gaze.

I.7 Contributions statement

James M. Rehg co-led the social benchmark effort and paper writing. Hanbyul Joo co-led the social benchmark effort and data annotation. Mike Zheng Shou co-led the social benchmark effort and problem formulation and modeling experiments. David Crandall led data collection at the Bloomington site and contributed to the social benchmark formulation and paper writing. Vamsi Ithapu contributed to the social benchmark formulation and data annotation. Hyun Soo Park led data collection at the Twin Cities site and contributed to the social benchmark formulation and paper writing.

Hao Jiang contributed to model development and data annotation. Yunyi Zhu contributed to model implementation and experiments. Eric Zhongcong Xu contributed to the social benchmark data preparation and the model implementation and experiments, and contributed to all data collection related tasks for the Singapore site. Ruijie Tao contributed to data collection for the Singapore site. Fiona Ryan led the data collection effort for the Atlanta site, including protocol design, multimodal sensor deployment and synchronization, and de-identification. Miao Liu contributed to data collection and analysis for the Atlanta site. Audrey Southerland contributed to the protocol design, IRB authoring and submission, participant recruiting, and data ingestion for the Atlanta site. Jayant Sharma contributed to participant recruiting, data collection, IRB submission, analysis, and data ingestion for the Twin Cities site. Yuchen Wang contributed to the protocol design, participant recruiting, and data collection for the Bloomington site. Weslie Khoo developed the multi-camera synchronization and de-identification pipelines at the Bloomington site.

Acknowledgements The social benchmark team would like to acknowledge the following additional contributions from individuals at each site: **Atlanta**: Jeffrey Valdez (recruitment and data collection), Gabriella Stripling, Ruth Stolovitz, and Andrea Sucre-Pardo (recruitment and dataset de-identification). **Twin Cities**: Reese Kneeland, Angad Cheema, Silong Tan, Anjali Oleksy, Zhiteng Cao, Diana Begelman (data collection and annotation) **Meta:** Samuel Clapp and Peter Dodds (binaural audio recording and multimodal synchronization). **Bloomington:** Zunaeed Salahuddin, Zehua Zhang, Ziwei Zhao.

J. Forecasting Benchmark

This section details the Forecasting benchmark task definitions, annotations, baseline models, and results.

J.1 Formal tasks definitions

As noted in the main paper, there are four forecasting tasks: future locomotion movement prediction, future hand prediction, short-term object interaction anticipation, and long-term action anticipation.

Future Locomotion Movements Prediction

This task aims to predict the future locomotion of a user given a sequence of past images. We formulate the problem as:

$$\mathcal{X} = \begin{bmatrix}\mathbf{x}_{t+1} & \cdots & \mathbf{x}_{t+F}\end{bmatrix}^\mathsf{T} = f(\mathbf{x}_{t-T}, \cdots, \mathbf{x}_{t-1}; \mathcal{I}), \quad (26)$$

where $\mathcal{X}$ is the future trajectory, T and F are the past and future time horizons, respectively, $\mathbf{x}_t$ is the point on the trajectory at time t, and $\mathcal{I}$ is the egocentric image at time t. With an assumption that the person walks over a major plane (e.g., ground plane), we represent the trajectory in a 2D plane, i.e., $\mathbf{x}_t \in \mathbb{R}^2$.

The essence of the locomotion task is to design a function f to predict a set of plausible K future trajectories $\{\mathcal{X}^k\}_k$ given the current image. Since there exists a number of plausible future trajectories with different topology, e.g., trajectories that bifurcate at an Y-junction, we predict K future trajectories.

Future Hand Prediction

In addition to future locomotion movements prediction, we consider another challenging task of predicting future hand positions of key-frames (see visual illustration in Fig. 50). Specifically, we denote the contact frame[17] as x_c, pre-condition frame[18] as x_p, and the three frames preceding the pre-condition frame by $0.5s$, $1s$ and $1.5s$ as x_{p_1}, x_{p_2}, x_{p_3}, respectively. Formally, given an input egocentric video $1.5s$ before the pre-condition time step (denoted as $x = \{x_{p_3-t_o-1}, ..., x_{p_3-1}\}$, with t_o referred as observation time), this task seeks to predict the positions of both hands (h_i^l, h_i^r) in the future key frames, where $i \in \{c, p, p_1, p_2, p_3\}$.

Short-Term Object Interaction Anticipation

This task aims to predict the next human-object interaction happening after a given timestamp. Given an input video, the goal is to anticipate:

- The spatial positions of the active objects, among those which are in the scene (e.g., bounding boxes around the objects). We consider the next active object to be the next object which will be touched by the user (either with their hands or with a tool) to initiate an interaction;
- The category of each of the detected next active objects (e.g., "knife", "tomato");
- How each active object will be used, i.e., what action will be performed on the active objects (e.g., "take", "cut");
- When the interaction with each object will begin (e.g., "in 1 second", "in 0.25 seconds"). This is the time to the first frame in which the user touches the active object (time to contact). This prediction can be useful in scenarios which involve human-machine collaboration. For instance, an assistive system could give an alert if a short time to action is predicted for a potentially dangerous object to touch.

In this task, models are required to make predictions *at a specific timestamp*, rather than densely throughout the video. Figure 51 illustrates the set-up. The model is allowed to process the video up to frame t, at which point it must anticipate the next active objects, and how they will take part in an interaction in δ seconds, where δ is unknown. The model can make zero or more predictions. Each prediction indicates the next active object in terms of noun class ($\hat{n}$) and bounding box ($\hat{b}$), a verb indicating the future action ($\hat{v}$), as well as the time to contact ($\hat{\delta}$), which estimates how many seconds in the future the interaction with the object will begin. Each prediction also comprises a confidence score ($\hat{s}$) used for evaluation.

Specifically, let V be an untrimmed video. We will denote with V_t the frame of V occurring at time-step t and with $V_{:t}$ the video segment starting at the beginning of V (timestamp 0) and ending at timestamp t. Given a timestamp t, denoted as "stopping time", the short-term object interaction anticipation task requires that a model is able to exploit the observed video $V_{:t}$ to predict N tuples (where N is arbitrary):

$$\{(\hat{b}_i, \hat{n}_i, \hat{v}_i, \hat{\delta}_i, \hat{s}_i)\}_{i=1}^N \quad (27)$$

where:

- $\hat{b}_i \in \mathbb{R}^4$ is a bounding box indicating the position of the predicted next active object;
- $\hat{n}_i \in \mathcal{N}$ is a *noun* indicating the class of the next active object, where $\mathcal{N}$ is the set of possible nouns.
- $\hat{v}_i \in \mathcal{V}$ is a *verb* indicating the action which will be performed, where $\mathcal{V}$ is the set of possible verbs;
- $\hat{\delta}_i \in R^+$ is the *time to contact*, a positive number which estimates how many second into the future the interaction with the object will begin;
- $\hat{s}_i \in [0, 1]$ is a confidence score associated to the prediction. Objects with a large confidence value are deemed to be likely next-active.

The model is allowed to perform N predictions for each observed example (with N arbitrary) both to account for the presence of multiple next-active-objects and to handle the multi-modality of future predictions. Each of the N predictions is intended as a plausible future object interaction. Figure 51 illustrates the proposed task. Given a video $V_{:t}$, a method should be able to detect the next active objects (e.g., two instances of "dough"), predict the action which will be performed with that object (e.g.,"take"), and the time to contact (e.g., $0.75s$).

Long-Term Action Anticipation

Long-term action anticipation aims to predict further into the future. Rather than predict the next action at a given timestamp, models

[17] The contact frame is defined as the first frame in which the user touches the object, hence the moment in which the object becomes active.

[18] As defined in Section G, the pre-condition frame marks a moment prior to the state-change of an object.

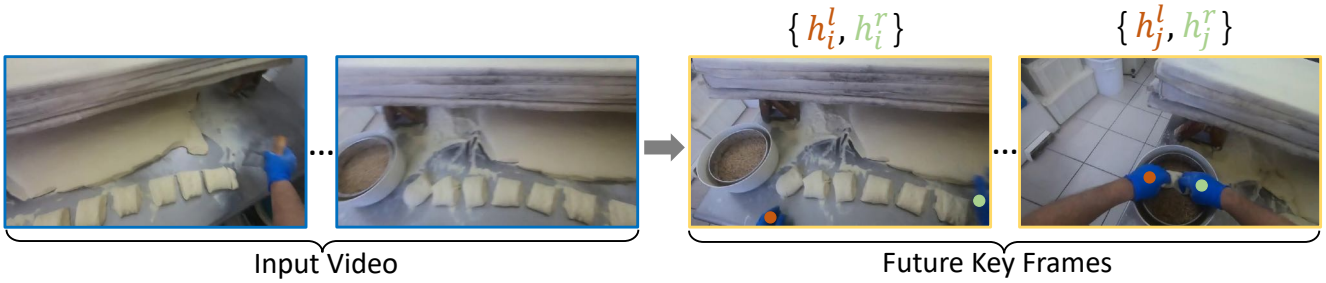


Figure 50. Example of future hand prediction.

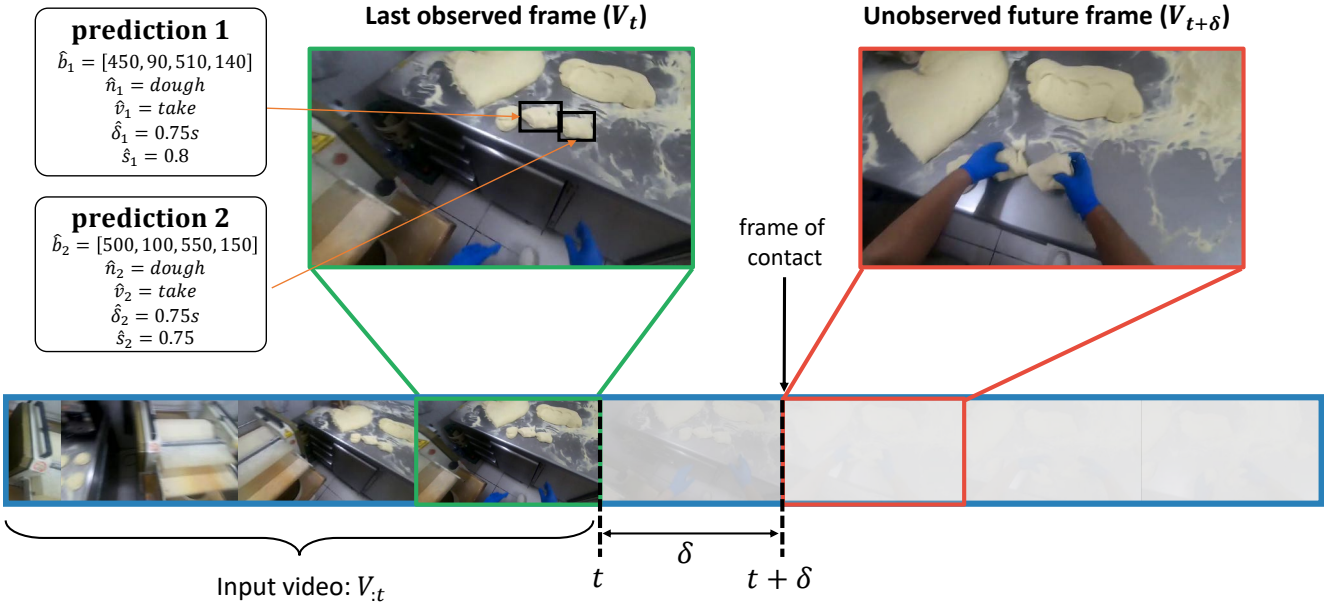


Figure 51. Example of short-term object interaction anticipation.

will be required to predict the sequence of Z future actions which the camera-wearer is likely to perform. This is important for long-horizon planning where a sequence of actions is required to be performed in a specific order to achieve a goal. Critically, these actions occur over long time horizons, may be of variable length and do not occur uniformly across time (e.g., an action every 5s). Thus, the task is defined at a more abstract level — models are required to predict sequences of action classes (verb and noun), rather than time to action or to next active objects bounding boxes in the current frame.

More formally, given an untrimmed video V and a *stopping time* t as described above, the long-term action anticipation model must observe $V_{:t}$ and predict N sets of sequences of Z plausible future actions:

$$\{\{(\hat{n}_{z,i}, \hat{v}_{z,i})\}_{z=1}^{Z}\}_{i=1}^{N} \tag{28}$$

where:

- $\hat{n}_{z,i} \in \mathcal{N}$ is the predicted noun and $\hat{v}_{z,i} \in \mathcal{V}$ is the predicted verb of the z-th future action.
- $\{(\hat{n}_{z,i}, \hat{v}_{z,i})\}_{z=1}^{Z}$ represents the sequence of future actions sorted by the predicted order in which they will appear in the video.

Like the short-term object interaction anticipation task, the model is allowed to generate N sets of predictions to account for the multi-modal nature of future prediction. Figure 52 illustrates the proposed task.

J.2 Data Selection

Future Locomotion Movements Prediction

Egocentric videos for locomotion and hand-object interaction are nearly mutually exclusive. Among these videos, we skim through

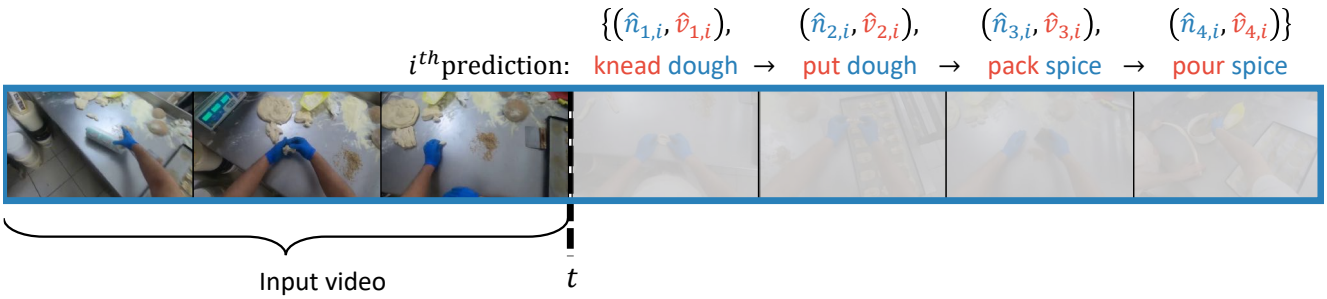


Figure 52. **Example of long-term action anticipation.** After observing a video up to a particular timestep t, a method should be able to predict the sequence of actions that will likely occur, in the correct order (e.g., first "take dough", next "put dough" etc.)

each video to manually identify video clips (beginning and end frames) that satisfy the following selection criteria. (1) Locomotion, by definition, involves diverse activities associated with walking. The clip should include substantial translational movement. (2) Each video clip must be longer than 10 seconds for past trajectory observation and future prediction. (3) The videos must observe surrounding scenes. This differs from the videos for hand-object interaction where the camera is deliberately tilted down to focus on the hand manipulation. We consider videos from glass-mounted cameras of which field of view approximately aligns with the first person. (4) 3D reconstruction and ground plane need to be accurate. After running structure from motion, we ensure 3D reconstruction from the videos achieves reasonable quality by checking 2D reprojection of the point cloud and ground plane. Given a set of these video clips, we choose frames for training/testing data for every second.

Remaining Tasks

For the remaining tasks we first manually ranked the scenarios based on their applicability to the forecasting tasks. For instance, scenarios like carpentery were high priority for forecasting whereas walking in the park was low-priority. We scored all scenarios from 1-3 based on this priority. We impose constraints on the minimum number of hours and participants to sub-select scenarios that have sufficient data for training (each participant should have contributed at least 15 minutes; and there should be at least 20 minutes of videos for that scenario). Next, we chunk our videos into 5 minute clips, and use the following algorithm to select clips to be labeled. To ensure geographical diversity, we distribute the total hours over universities and randomly select clips from each to fill the hours allocated to that university. If there are universities that contributed less, then their hours are distributed across the other universities. To select the clips given a university and the hours allocated; we would first sample a participant, then sample a video for that participant, and sample 1 clip from that video. For certain repetitive scenarios (like brick making), we reject this clip if we already have selected at least 2 clips from the same video. We repeat the process until the required number of hours are selected.

Data	Outdoor	Indoor	Mixed	Total
Train	34.1k	0.41k	16.7k	51.3k
Val	7.5k	0.23k	6k	13.9k
Test	7.4k	0.18k	3k	10.6k

Table 32. We split the image data for locomotion prediction based on scenes that including outdoor, indoor, and mixed.

J.3 Data Annotation

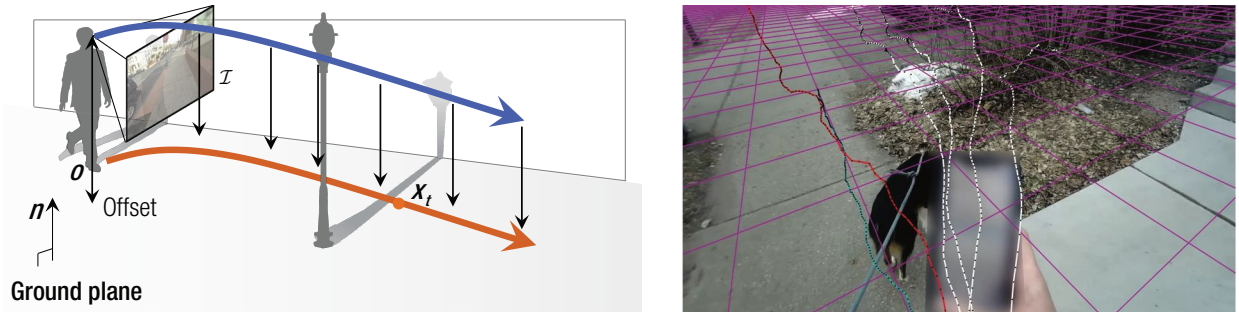
Future Locomotion Movements Prediction

We generate the ground truth of future trajectories using 3D reconstruction of the camera trajectories. Given a sequence of egocentric images, we reconstruct the 3D egomotion and scene geometry using a standard structure from motion pipeline with a few modification to handle a large number of images. With the 3D scene point cloud, we estimate the ground plane using RANSAC with the ground plane normal prior. The 3D reconstructed camera trajectory is projected onto the ground plane to form the 2D future trajectory as shown in Figure 53.

Our image dataset includes locomotion in outdoor, indoor, and mixed scenes. We split the image data into training/validation/testing sets with approximately 70%/15%/15%, respectively. The ratio across scenes does not exactly match because the split is performed based on the (anonymous) participant ID. The summary of the data split can be found in Table 32.

Future Hands Movements Prediction

For the the future hand position and trajectory prediction, the annotation will be performed by labeling bounding boxes around hands in the frame in which the user touches the active objects as well as in frames preceding each object interactions. Hands bounding boxes will be associated to a label useful to distinguish among left and right hands. Therefore, given an object interaction, we will annotate key frames preceding the beginning of the interaction. Specifically, t_c and t_p denote the time step of contact frame and pre-condition frame, and $t_{p_1}, t_{p_2}, t_{p_3}$, denote time steps $0.5s$, $1s$ and $1.5s$ before the pre-condition time step. Therefore, for each interaction there will be 5 key frames labeled with bounding boxes of hands, including the contact frame. We use the bouding box center as the ground truth of hands positions.



(a) Geometry

(b) Future trajectory prediction

Figure 53. (a) We represent the future trajectory of a person using the ground plane. Given the 3D reconstruction of the camera trajectory, we project it into the estimated ground plane to form the future trajectory. (b) The ground truth future trajectory (blue) and the predicted trajectories (red and white) are shown in the egocentric image with the ground plane coordinate (magenta grid). We predict top 5 trajectories where the top prediction is marked in red.

Short-Term Object Interaction Anticipation

Each video V of the dataset is labeled with a set of short term object interaction anticipation annotations $\mathcal{S}_V = \{S_V^{(j)}\}_j$ indicating the occurrence of object interactions in the video. Each annotation

$$S_V^{(j)} = (t_s^{(j)}, \{n_h^{(j)}\}_h, v^{(j)}, \{A_h^{(j)}\}_h, \{B_h^{(j)}\}_h) \quad (29)$$

includes:

- $t_s^{(j)}$: the timestamp indicating the beginning of the interaction with the active objects. This is the first frame in which the user touches at least one of the active objects;
- $\{n_h^{(j)}\}_h$: the set of categories of the h interacted objects;
- $v^{(j)}$: the class of the action involving the active objects;
- $\{A_h^{(j)}\}_h$: the bounding box annotations for the active objects. The cardinality of $\{A_h^{(j)}\}_h$ is equal to the cardinality of $\{n_h^{(j)}\}$, i.e., $|\{A_h^{(j)}\}_h| = |\{n_h^{(j)}\}|$. The h^{th} set $\{A_h^{(j)}\}_h$ contains bounding box annotations for the active objects of category n_h at timestamp $t_s^{(j)}$;
- $\{B_h^{(j)}\}_h$: the bounding box annotations for the next active objects. The cardinality $\{B_h^{(j)}\}_h$ is equal to the cardinality of $\{A_h^{(j)}\}_h$, i.e., $|\{B_h^{(j)}\}_h| = |\{A_h^{(j)}\}_h|$. The j^{th} set $B_h^{(j)}$ contains the bounding box annotations of next active objects of class n_h. In particular, $B_h^{(j)}$ contains annotations for the same object instances annotated in $A_h^{(j)}$, tracked in frames preceding $t_s^{(j)}$. Specifically, $B_h^{(j)} = \{B_{l,h}^{(j)} | l = 1, ..., m\}$, where $B_{l,h}^{(j)}$ is the set of bounding box annotations of next active object of class n_h annotated at timestamp $t_s - l\alpha$. Here m indicates the number of frames preceding the beginning of the interaction in which objects are annotated, whereas α is the temporal distance between the sampled frames. For instance, setting $\alpha = 0.5s$ and $m = 4$, we will label the frame in which the object is interacted as well as 4 frames in a $2s$ segment preceding the interaction. Figure 54 shows an example of how frames are sampled with the considered scheme.

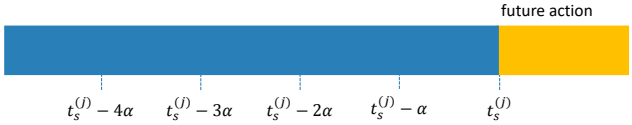


Figure 54. An example of how frames are sampled to be labeled with next active object annotations. For a given action i, we sample m frames at regular intervals α. If we set $m = 4$ and $\alpha = 0.5$, we label the frame of contact as well as 4 frames along a segment of $2s$ preceding the beginning of the action at a framerate of $2fps$.

Figure 55 reports a sample clip with the discussed annotations. The timestamp t_s is selected as the first one in which the user touches the active objects. The frames following this timestamp are not labeled. Active object bounding boxes are labeled at timestamp t_s, whereas next active object bounding boxes are labeled in frames preceding t_s.

Long-Term Action Anticipation

Each video V is labeled with a set of long-term action annotations $\{L_V^{(j)}\}_j$, corresponding to a *stopping time* until which the video can be observed, and a sequence of Z future action labels defined as follows:

$$L_V^{(j)} = (t^{(j)}, \{(n_z^{(j)}, v_z^{(j)})\}_{z=1}^{Z}) \quad (30)$$

where:

- $t^{(j)}$: the timestamp until which the video can be observed (i.e., $V_{:t^{(j)}}$) before making predictions of future actions;
- $n_z^{(j)}$: the noun category of the primary interacted object in the z-th future action;
- $v_z^{(j)}$: the verb describing how the objects will be interacted with in the z-th future action .

For each video, $t^{(j)}$ are selected from the *last* timestamp of each annotated object interaction. It is worth noting that once short-term annotations $S_V^{(i)}$ are available (see Section J.3) and a value for Z has been chosen, the long-term annotations $L_V^{(j)}$ can be easily

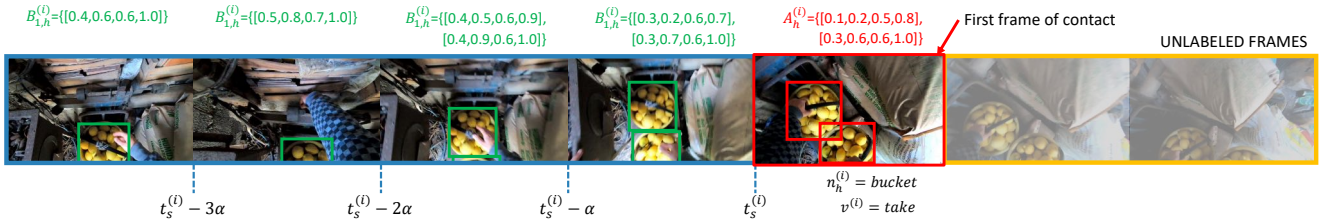


Figure 55. Example of annotations for the short-term object interaction anticipation task.

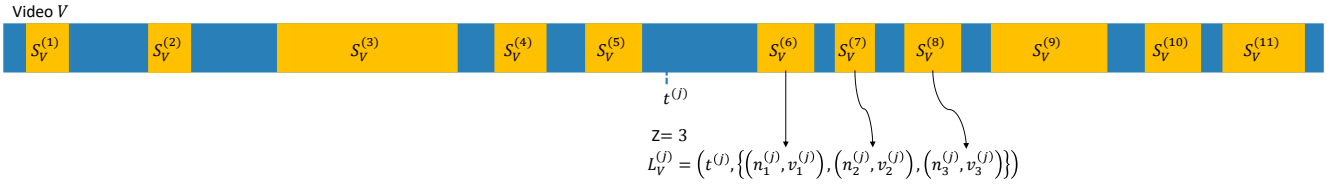


Figure 56. An example of a long-term annotation $L^{(V,t)}$ for an untrimmed video V at timestamp t can be obtained from short-term annotations $S_V^{(i)}$. In the example, $Z = 3$, hence the long term annotation is obtained by considering the first three actions beginning after timestamp t.

obtained by sampling the first Z actions annotated in video V beginning after timestamp $t^{(j)}$. More formally, the future action labels for $L_V^{(j)}$ are obtained as:

$$\{(n_0^{(i_z)}, v^{(i_z)}) \,|(t_s^{(i_z)}, \{n_h^{(i_z)}\}_h, v^{(i_z)}, \{A_h^{(i_z)}\}_h, \{B_h^{(i_z)}\}_h) \in \mathcal{S}_V \wedge$$
$$t_s^{(i_z)} \geq t^{(j)} \wedge$$
$$t_s^{(i_1)} \leq \ldots \leq t_s^{(i_Z)} \wedge$$
$$\nexists S_V^{(j)} \in \mathcal{S}_V | t^{(j)} \notin \{i_1, \ldots, i_Z\}, t \leq t_s^{(j)} < t_s^{(i_Z)}\}_{z=1}^{Z}$$

where $n_0^{(i_z)}$ refers to the primary interacted object from the set of interacted objects $\{n_h^{(i_z)}\}_h$. Figure 56 illustrates an example of how long-term annotations are obtained form short-term annotations.

Annotation analysis

Dataset statistics As discussed earlier, one of our primary objectives when selecting the data to annotate was to maximize the diversity in terms of activities and geographic locations. Our dataset includes scenarios spanning a wide range of everyday activities (*e.g.*, gardening, cleaning, fishing, *etc.*). In addition to diversity across scenarios, there is also geographic diversity within scenarios. For example, cooking may look very different in Italy, India, Saudi Arabia, or Japan. In Figure 38, we show the resulting scenario and university distributions. Overall, our benchmark consists of 120 hours of annotated video coming from 53 scenarios, 7 universities, and 406 participants.

Temporal structure of activities Human activity is goal-driven and structured over time, with certain action sequences being favored over others. We measure this temporal structure using Normalized Pointwise Mutual Information (NPMI) [41] over pairs of actions following prior work [92]. NPMI is a measure of how likely actions follow each other. In our dataset, typical patterns include "pull grass → throw grass (0.87)", "hold spinach → cut spinach (0.83)", "turn-on faucet → turn-off faucet (0.68)", "take cloth → fold cloth (0.49)" etc. Several actions also occur in sequence with high NPMI scores due to the repetitive nature of the activity. For example, "flip page → flip page (0.83)" while reading, or "cut carrot → cut carrot (0.82)" while cooking. Finally, we see common action sequences involving multiple objects like "fill tire → close valve (0.89)", or "lift vacuum-cleaner → clean staircase (0.87)". This structure is valuable and can inform long-term action anticipation models.

Dataset split To facilitate future research and comparisons, we construct training, validation, and test splits containing 40%, 30%, and 30% of the data, respectively. We note, however, that we do not release the ground truth annotations for the test set. Following common practice, evaluation on the test set will be supported through the public evaluation server and leader board. We assign data to splits randomly at the level of 5 minute clips. This ensures that all interactions within a 5 minute clip were labeled by an annotator and provides enough temporal context for long-term video tasks, like long-term action anticipation.

J.4 Evaluation measures

Future Locomotion and Hands Movements Prediction

Future Locomotion We measure the accuracy of the prediction using two metrics. (1) K best mean trajectory error (K-MTE): we measure K best trajectory error:

$$\text{K} - \text{MTE} = \underset{\{\mathcal{X}_k\}_{k=1}^{K}}{\text{argmin}} \frac{1}{\sum_t v_t} \sum_t v_t \|\mathbf{x}_t - \widehat{\mathbf{x}}_t\|, \qquad (31)$$

$\mathbf{x}_t \in \mathbb{R}^2$ is the predicted location at time t, $\widehat{\mathbf{x}}_t$ is the ground truth location, and v_t is the visibility. The visibility indicates the availability of the ground truth trajectory, i.e., due to severe

egocentric videos, the ground truth trajectories may include missing data. $v_t = 0$ indicates missing data at time t. (2) Probability of correct trajectory (PCT): we measure the success rate of the correct trajectory retrieval:

$$\text{PCT}\epsilon = \frac{1}{K}\delta\left(\frac{1}{\sum_t v_t}\sum_t v_t\|\mathbf{x}_t - \widehat{\mathbf{x}}_t\| < \epsilon\right), \quad (32)$$

where $\delta(\cdot)$ is one if the statement is true and zero otherwise. ϵ is the trajectory error tolerance, i.e., if the trajectory error is smaller than the error tolerance, it is considered as a correct trajectory prediction. $\text{PCT}\epsilon$ measures how many trajectories among K retrieved trajectories are close to the ground truth trajectory.

Future Hand Movement As for the future hands movements prediction, we only consider the key frame prediction, and therefore adopt Mean Key Frame Displacement Error Contact (M.Disp.) Key Frame Displacement Error as evaluation metrics (C.Disp.):

- Mean Key Frame Displacement Error (M.Disp.):

$$D_m = \frac{1}{n}\sum_{i\in H_t}\|h_i - \hat{h}_i\| \quad (33)$$

  H_t refers to the set of visible hand positions of key frames, and n is the length of set H_t. h_i denotes the predicted hand position in the image coordinate, while $\hat{h}^i$ denotes the ground truth hand positions.

- Contact Key Frame Displacement Error (C.Disp.):

$$D_c = \|h_c - \hat{h}_c\| \quad (34)$$

  h_c refers to the hand positions at Contact frame.

Note that all reports are reported on downsampled video frames with height of 256 and original aspect ratio.

Short-Term Object Interaction Anticipation

Methods will be evaluated at the timestamps in which next-active objects have been annotated, i.e.,

$$\Big\{t|t \;=\; t_s - l\cdot\alpha \quad \forall t_s \in \{t_s^{(j)}|\exists\overline{h} : B_{\overline{h}}^{(j)} \neq \emptyset\}_j \quad \forall l \in \{1,...,m\}\Big\} \quad (35)$$

where $\{t_s^{(j)}|\exists\overline{h} : B_{\overline{h}}^{(j)} \neq \emptyset\}_j$ is the set of all timestamps indicating the beginning of an interaction, for which at least one next active object has been annotated, and α and m are defined in Appendix J.3.

Since detecting next active objects is a major part of the task, we base our evaluation measures on mean Average Precision (mAP), as defined in the Pascal VOC challenge [60]. As in standard mAP, we first match each of the detected next active objects to ground truth annotations. A predicted and a ground truth bounding boxes are a possible match if their Intersection Over Union (IOU) value exceeds 0.5 and if some matching criteria are met. We will define matching criteria later. Predictions are matched to ground truth annotations belonging to the same evaluated example in a greedy fashion, prioritizing predictions with higher confidence scores and choosing matches corresponding to larger IOU values. A ground truth annotation can be matched at most with one predicted box. All matched predictions are counted as true positives, whereas all unmatched predictions are counted as false positives. Performance on the whole test set is summarized using the mean of the Average Precision values obtained for each class.

To account for the multi-modal nature of future predictions (i.e., more than one next active object can be likely), we "discount" the number of false positives obtained in a given example by the number of available ground truth annotations in that example multiplied by $K-1$, where K is a parameter of the evaluation measure. Specifically, if an example contains two ground truth annotation, we ignore the $(K-1)*2$ false positives with the highest scores. This effectively implements a "Top-K mean Average Precision" criterion which does not penalize methods for predicting up to $K-1$ possibly likely next active objects which are not annotated. Given a generic prediction $(\hat{b}_i, \hat{n}_i, \hat{v}_i, \hat{\delta}_i\hat{s}_i)$ and a generic ground truth annotation $(b_j, n_j, v_j, \delta_j)$, we define the following variants of this Top-K evaluation measure considering different matching criteria:

- Noun Top-K mAP: prediction i and annotation j are a possible match if the following conditions are satisfied:
  - $IOU(\hat{b}_i, b_j) > 0.5$;
  - $\hat{n}_i = n_j$;
- Noun + Verb Top-K mAP: prediction i and annotation j are a possible match if the following conditions are satisfied:
  - $IOU(\hat{b}_i, b_j) > 0.5$;
  - $\hat{n}_i = n_j$;
  - $\hat{v}_i = v_j$.
- Noun + TTC Top-K mAP: prediction i and annotation j are a possible match if the following conditions are satisfied:
  - $IOU(\hat{b}_i, b_j) > 0.5$;
  - $\hat{n}_i = n_j$;
  - $|\hat{\delta}_i - \delta_j| < T_\delta$.
- Overall Top-K mAP: prediction i and annotation j are a possible match if the following conditions are satisfied:
  - $IOU(\hat{b}_i, b_j) > 0.5$;
  - $\hat{n}_i = n_j$;
  - $\hat{v}_i = v_j$;
  - $|\hat{\delta}_i - \delta_j| < T_\delta$.

  Where T_δ is a tolerance threshold, parameter of the evaluation measure.

The goal of the different measures is to assess the ability of the model to predict next object interactions at different levels of granularity. We use $K = 5$ and $T_\delta = 0.25$.

Long-Term Action Anticipation

Methods will be evaluated at the set of timestamps specified by the end of each annotated object interaction in a video V. Let $L_V^{(j)} = \{(n_z^{(j)}, v_z^{(j)})\}_{z=1}^{Z}$ be the ground truth annotation related to video V at time-stamp $t^{(j)}$ and let $\{\{(\hat{n}_{z,k}^{(j)}, \hat{v}_{z,k}^{(j)})\}_{z=1}^{Z}\}_{k=1}^{K}$ be the K predicted sequences of Z actions. We will consider single noun/verb/action predictions correct following the definitions discussed in Section J.4. The K predicted sequences will hence be evaluated using the edit distance metric as follows.

For a given k, this is obtained by evaluating the edit distance between a predicted sequence and the ground truth sequence of future actions. The edit distance

$$\Delta_E(\{(\hat{n}_{z,k}^{(j)}, \hat{v}_{z,k}^{(j)})\}_{z=1}^{Z}, \{(n_z^{(j)}, v_z^{(j)})\}_{z=1}^{Z})$$

is computed as the Damerau-Levenshtein distance [47, 133] over sequences of predictions of verbs, nouns and actions. The goal of this measure is to assess performance in a way which is robust to some error in the predicted order of future actions. A predicted verb/noun is considered "correct" if it matches the ground truth verb label at a specific time-step. The allowed operations to compute the edit distance are insertions, deletions, substitutions and transpositions of any two predicted actions. Following the "best of many" criterion, the K predictions are evaluated considering the smallest edit distance between the ground truth and any of the K predictions:

$$\Delta_E(\{\{(\hat{n}_{z,k}^{(j)}, \hat{v}_{z,k}^{(j)})\}_{z=1}^{Z}\}_{k=1}^{K}, \{(n_z^{(j)}, v_z^{(j)})\}_{z=1}^{Z}) = \min_{k=1..K} \Delta_E(\{(\hat{n}_{z,k}^{(j)}, \hat{v}_{z,k}^{(j)})\}_{z=1}^{Z}, \{(n_z^{(j)}, v_z^{(j)})\}_{z=1}^{Z})$$

Note that we consider edit distance over simple accuracy based measures. Treating predictions for each future time-step independently and calculating accuracy does not account for the sequential nature of the prediction task where the order of predictions is important. We evaluate each metric independently for verbs, nouns and actions (verb and noun together). We report edit distance at $Z = 20$ (ED@20) and use $K = 5$ in our experiments. We select $Z = 20$ as baselines begin to predict actions at random for higher values of Z.

J.5 Baseline definitions and implementation details

Future Locomotion Movements Prediction

We make use of the method by Park et al. [175] for a baseline algorithm. The method models the trajectory prediction function in Equation (26) using KNN classification with CNN image encoding, i.e.,

$$\{\mathcal{X}\} = KNN\left(\{\phi(\mathcal{I}_i)\}, \phi(\mathcal{I})\right) \tag{36}$$

where $KNN(A, B)$ finds the K nearest neighbor of B given the set A, and $\phi(\mathcal{I}) \in \mathbb{R}^n$ is a function that extracts the image feature of $\mathcal{I}$. We use the AlexNet image feature extractor for ϕ.

Notably, the baseline algorithm leverages a polar coordinate system to represent the trajectory, i.e., $\mathbf{X}_j^{2D} = \begin{bmatrix} r_j & \theta_j \end{bmatrix}^\mathsf{T}$ is a 2D trajectory on the ground plane where r_i and θ_i are the polar coordinates of the trajectory represented in the egocentric coordinate system, i.e., distance (radial) and direction (angle) with respect to the person's feet location as shown in Figure 53:

$$\mathbf{X}_j^{2D} = \texttt{cart2polar}(\mathbf{r}_1^\mathsf{T}\mathbf{X}_j, \mathbf{r}_2^\mathsf{T}\mathbf{X}_j) \tag{37}$$

where $\mathbf{r}_1$ and $\mathbf{r}_2$ are the two spanning vectors of the ground plane that are aligned with the rotation matrix $\mathbf{R}_t$. $\mathbf{r}_1$ is the facing direction and $\mathbf{r}_2$ is lateral direction. Both are perpendicular to the ground plane normal $\mathbf{n}$ as shown in Figure 53. `cart2polar` is a coordinate transform from cartesian to polar coordinates.

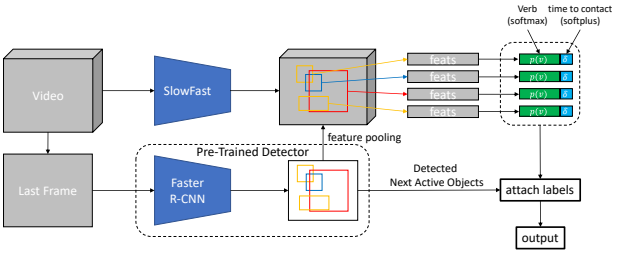


Figure 57. Short-Term object interaction anticipation baseline.

Future Hands Movements Prediction

Baseline Description The proposed future hand movement prediction task can be factorized as a regression problem. To address this task, we adopt a baseline that utilizes the I3D network as the backbone to extract the spatial-temporal video representations of the input video sequence, and then use a linear mapping function as the regressor to predict the future keyframe hand positions. We adopt the smoother l1 loss as the objective function:

$$L_h = \begin{cases} 0.5 * w * (h - \hat{h})^2/\beta, & \text{if } |h - \hat{h}| < \beta \\ w * (|h - \hat{h}| - 0.5 * \beta), & \text{otherwise} \end{cases} \tag{38}$$

where $h \in R^{20}$ is a vector that represents the x,y coordinates of both left and right hands in the aforementioned five future key frames. If the hand is not observed in the keyframe, we pad 0 into the $\hat{h}$, and adopt a binary mask w to prevent the gradients propagation of these unobserved instances.

Training Details We adopt the I3D model as the backbone network and a regression header, composed of two linear operations, to predict the hand positions in the future key frames. For our experiments, we set observation time T_o as 2s. For training, we applied several data augmentation techniques, including random flipping, rotation, cropping and color jittering to avoid overfitting. Our baseline model was trained with a batch size of 64 for 25 epochs using a cosine learning rate decay with a initial learning rate of 0.0375. We set β to 5 in the weighted smoothed L1 loss as introduced in Eq. 38.

Short-Term Object Interaction Anticipation

Data and annotations used for the experiments We performed our experiments on a subset of the data and annotations to obtain verb and noun taxonomies consistent with the Short-Term

Object-Interaction Anticipation task. We started by considering all annotated actions for which a contact frame has been specified by the annotators. Note that these constitute about 30% of the whole set of annotated actions and that the notion of a contact frame is fundamental to our task. We then gathered all annotated frames and referenced them to their respective contact frames, computing the time to action targets. We discarded all those annotations which comprised a verb or a noun class marked by the annotator as "null". We further discarded annotations related to nouns which had been labeled inconsistently and non-object classes such as "wall" or "wallpaper". We similarly removed all annotations related to the verb "talk" which do not involve interactions with objects.

To avoid having an over-specific noun taxonomy, we clustered selected noun classes into homogeneous groups. For instance the nouns "okra", "apple", "celery" and "avocado" have all been grouped under the "vegetable_fruit" class. We also grouped verbs which have similar semantic when anticipated. For instance, the verbs "take", "carry", "lift", "pull" and "remove" have all been grouped in the "take" cluster. Note that while these actions may be visually different, they all have similar effects on objects, which makes them indistinguishable when anticipated. We further removed all annotations related to nouns appearing less than 50 times in the test set (we follow the common split defined for this benchmark). We choose to retain only nouns appearing at least 50 times in the test set to allow for a reliable evaluation through the mAP measure.

The final set of data includes $64,798$ annotated examples in total with 87 nouns and 74 verbs. Our taxonomy is adapted from the one presented in Figure 39. Figure 58 and Figure 59 report the distributions of verb and noun annotations in the selected data. Among the $64,798$ annotations, $27,801$ are in the training set, $17,217$ are in the validation set, and $19,780$ are in the test set.

Baseline Description Figure 57 illustrates the proposed baseline for short-term object interaction anticipation. The baseline includes two main components. A Faster R-CNN object detector [87] is used to detect next active objects in the last frame of the input video clip processed at full resolution. A SlowFast 3D CNN [71] is hence used to predict a verb label and a time to action for each predicted object. This is done by obtained a fixed-length representation of each object through ROI pooling [87]. Two linear layers are hence used to predict a probability distribution over verbs and a positive quantity for time to contact prediction respectively. Verb probability distributions are obtained using a softmax layer, whereas a softplus activation is used for time to contact prediction to make sure that the prediction is a positive number. The final output of the model is obtained by attaching the predicted verb and time to contact to each detected next active object. The noun label and confidence scores are copied from the output of the Faster R-CNN component.

Training Details We first train the Faster R-CNN component on all frames with annotated next active objects. We use the Faster RCNN detector based on ResNet50 using the "3x" training schedule provided with the Detectron2 library[19]. After this stage, the weights of the Faster R-CNN component are not updated anymore. We hence train a SlowFast model based on ResNet50. We follow the configuration provided in the PySlowFast library[20] to tackle the AVA detection task ("SLOWFAST_32x2_R50_SHORT.yaml"). The SlowFast model takes as input video clips of 32 frames sampled with a temporal stride of 1 frame. During training, we match each detected object to the ground truth instance with largest Intersection Over Union (IOU), provided that it is larger than 0.5. We hence attach the verb and time to contact labels of the ground truth boxes to the matched ones. We then train the model applying the following loss only to boxes which have been matched to ground truth instances:

$$\mathcal{L} = \mathcal{L}_v + \lambda \mathcal{L}_{ttc} \tag{39}$$

where $\mathcal{L}_v$ is the cross entropy loss for verb prediction, $\mathcal{L}_{ttc}$ is the smooth L1 loss [87] applied to time to contact prediction, and we set $\lambda = 10$ to control the contributions of the two losses. To regulate the number of frames processed by the slow branch, we set $\alpha = 8$. We train the model on 4 NVIDIA V100 GPUs with a batch size of 64 for 50 epochs using a cosine learning rate policy with a base learning rate of 0.001. We validate the model at the end of each epoch and consider the weights which achieved the best overall top-5 mAP on the validation.

Long-Term Action Anticipation

Baseline Description The goal of the baseline model is to take as input a trimmed video of arbitrary length, and predict N different plausible sequences of future actions. The baseline models thus consist of three components: (1) the encoder backbone for obtaining clip level features, (2) the aggregation module for combining the obtained features from different clips, and (3) the decoder network for decoding the plausible sequences of future actions. For encoder backbones, we consider state of the art video recognition networks from both convolutional model, namely, SlowFast [71] and the newly proposed video transformer models, namely, MViT [63]. For aggregation module, we experiment with simple concatenation operators that concatenates the obtained clip features from multiple input clips as well as transformer based self-attention modules. For the decoder networks we consider the following options:

- No Change: A simple recognition baseline that assumes no future change in the current action and simply predicts the *currently* observed action as a duplicated static future sequence for Z steps.
- MultiHead: This model trains Z independent heads in parallel, one for each future time step. The final sequence is simply the conjoined predicted actions of each head.

Finally, to generate N plausible future sequences for constructing multimodal baselines, we simply sample the predicted future action distribution N times. The framework for a particular instantiation of the MultiHead baseline is illustrated in Figure 60.

Training Details For each video, we sample multiple input clips to process with our backbone network. A single clip length for both the backbones, SlowFast and MViT, comprises of 16 frames sampled 4 frames apart. Each clip is processed independently by the same encoder weights and combined with the aggregation module. The aggregated feature is decoded with the decoder module where

[19] https://github.com/facebookresearch/detectron2

[20] https://github.com/facebookresearch/SlowFast

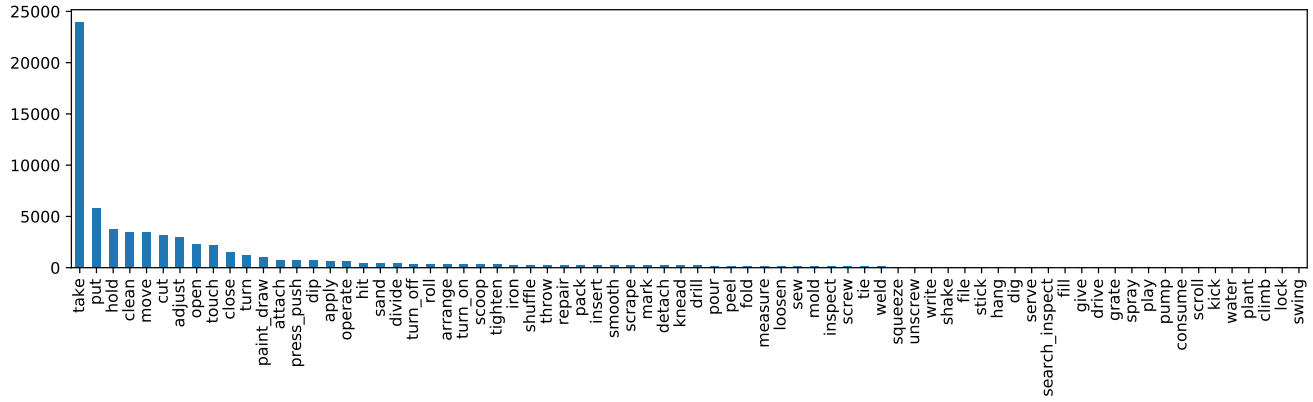


Figure 58. Verb distribution in the Short-Term Object-Interaction Anticipation data.

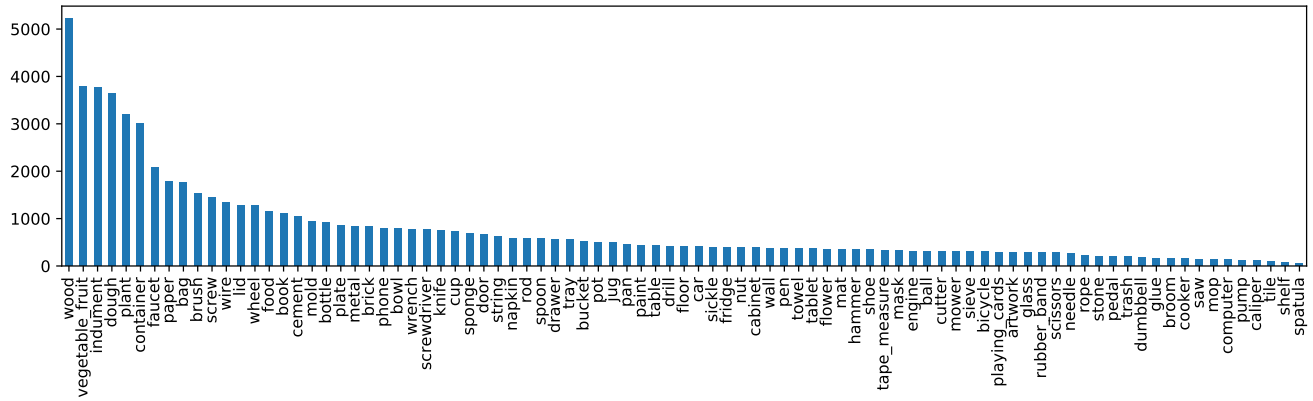


Figure 59. Noun distribution in the Short-Term Object-Interaction Anticipation data.

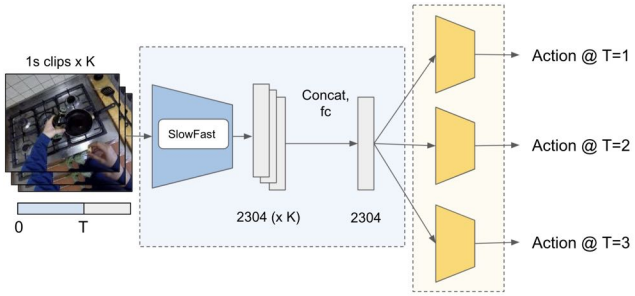


Figure 60. **Long-Term Action Anticipation baseline**. A baseline model with a SlowFast backbone, and $Z = 3$ is shown here. Blue box: clip encoder network. Yellow box: multiple classifier heads, one for each future action. See Sec. J.5 for more details.

the output behavior changes during training and testing. In training, the decoder predicts the next action probability distributions for each future step. We calculate the sum of losses for each prediction as our total loss:

$$\mathcal{L}_{lta} = \sum_{z=1}^{Z} \mathcal{L}_v((p_z^n, p_z^v), (n_z, v_z)) \tag{40}$$

where $\mathcal{L}_v$ is cross entropy loss, p_z^* refers to the predicted probability distribution over verbs and nouns, and (n_z, v_z) refer to the ground truth future action labels.

During testing, we sample action class labels $(\hat{n}_z, \hat{v}_z)$ from the predicted distribution independently for each future step. We repeat this sampling procedure N times to generate multiple cancidate sets of predictions for evaluation described in Section J.4.

We use the taxonomy presented in Figure 39 for our experiments. We finetune a Kinetics-400 [109] pretrained encoder backbones on Ego4D action recognition and use this model for all baselines to extract the clip level features. The aggregation module and decoder networks are trained from random initialization directly on the forecasting task. The encoder weights are kept unchanged during the decoder network training. We set $Z = 20$ for long horizon future evaluation and $K = 5$ as the number of plausible future sequences predicted by the model. For all baselines, we sample 2 input clips to capture past context unless otherwise specified. We train the model on 8 NVIDIA V100 GPUs with a batch size of 64

Set	Metric	Mean	Median
Val	5-MTE	5.11m	2.53m
Val	3-MTE	6.19m	2.99m
Val	1-MTE	8.81m	4.63m
Test	5-MTE	4.84m	2.69m
Test	3-MTE	5.54m	3.24m
Test	1-MTE	7.66m	4.73m

Table 33. Results of the locomotion prediction task. We report mean/median for 7-15 second predictions. We use $K = 1, 3, 5$.

Set	$\epsilon = 1m$	$\epsilon = 2m$	$\epsilon = 3m$	$\epsilon = 4m$	$\epsilon = 5m$	$\epsilon = 6m$
Val	0.14	0.29	0.39	0.46	0.51	0.54
Test	0.16	0.31	0.40	0.47	0.53	0.58

Table 34. Results of the locomotion prediction task. We report the probability of correct trajectory (PCT) as varying the error threshold ϵ.

Set	Method	Left Hand		Right Hand	
		M.Disp.↓	C.Disp.↓	M.Disp.↓	C.Disp.↓
Val	I3D+Reg	54.11	57.29	54.73	57.94
Test	I3D+Reg	52.98	56.37	53.68	56.17

Table 35. Results of future hand movement prediction task. Note that the left and right hands movements are evaluated separately. ↓ indicates lower is better

for 30 epochs and a base learning rate of 0.0001.

J.6 Results

Future Locomotion Movements Prediction

We evaluate the KNN based baseline algorithm by measuring mean trajectory error (K-MTE) and probability of correct trajectory (PCT) given an error tolerance. The trajectory length ranges from 7 to 15 seconds (70-150 points in a trajectory given 10 FPS). Our baseline achieves mean error 8.81m for $1-\text{MTE}$ and 0.39 for $\text{PCT}_{\epsilon=3m}$. The result is summarized in Table 33 and 34.

Future Hands Movements Prediction

For future hands movements prediction task, we report mean displacement error (M.Disp.) and contact frame displacement error (C.Disp.) on both validation and test sets in Table 35. Our baseline model achieves M.Disp. of $(52.98/53.68)$ and C.Disp. of $(56.37/56.17)$ for left/right hand position prediction on the test set. It is worth noting that predicting hand positions on contact frame is more challenging than on other key frames. This is because, by the definition of contact frame and pre-condition frame, the anticipation temporal footprint of contact frame is larger than other key frames. We further provide qualitative results of our baseline method in Fig. 61. Notably, the model can make reasonable predictions on future hand positions. However, the model is more likely to fail when there is drastic embodied motions.

Set	Method	Noun	Noun+Verb	Noun+TTC	Overall
Val	FRCNN+Rnd.	17.55	1.56	3.21	0.34
Val	FRCNN+SF	17.55	5.19	5.37	2.07
Test	FRCNN+Rnd.	20.45	2.22	3.86	0.44
Test	FRCNN+SF	20.45	6.78	6.17	2.45

Table 36. Results of the short-term object interaction anticipation task. See text for discussion.

Short-Term Object Interaction Anticipation

Table 36 reports the results for the short-term object interaction anticipation task on both the validation and test sets. We compare the proposed baseline based on Faster RCNN and SlowFast ("FRCNN+SF" in the table) with a simpler baseline which uses Faster RCNN to detect object and predict their classes, but draws verb and TTC predictions randomly from the training set distribution ("FRCNN+Rnd." in the table). Results are reported in Top-5 mAP% according to the different matching criteria discussed in Appendix J.4. As can be noted, the proposed baseline outperforms random prediction by big margins when verbs and TTCs are predicted on both the validation and test sets. This suggests that, despite being simple, the baseline can leverage the observed video to anticipate future object interactions. Figure 62 reports some qualitative examples of the baseline. The model is sometimes able to detect the next active objects and predict suitable verbs and TTCs, but performance tends to be limited especially in complex scenarios.

Long-Term Action Anticipation

Table 37 shows our results on both the validation and test sets. The *No Change* baseline simply predicts the current action as the next Z actions, and performs poorly at predicting future actions. Explicitly training multiple heads improves performance on verbs, nouns and actions. Changing the backbone architecture from SlowFast to MViT greatly improves verb forecasting prediction performance, but deteriorates noun forecasting performance, highlighting the trade-off between the two despite similar action classification performance on Kinetics. Finally, including larger video context information in the form of multiple input clips by using the transformer based aggregator module results in the best performance.

Figure 63 shows some qualitative results of our method. In each row, the ground truth future actions are shown along with the predictions from our model (for 5 time-steps). Correct predictions are highlighted in green, while valid actions that are incorrectly ordered (or paritally correct) are highlighted in blue. Note that though not perfectly aligned, incorrectly ordered sequences are given partial credit via the edit-distance metric.

J.7 Discussion

Data Annotation

Annotating the videos for forecasting tasks posed a number of interesting challenges. First, we found the diversity of the data led to a large and diverse taxonomy, which some annotators found hard to navigate. Hence, we found a number of annotators used the "OTHER" option, which we eventually manually mapped to

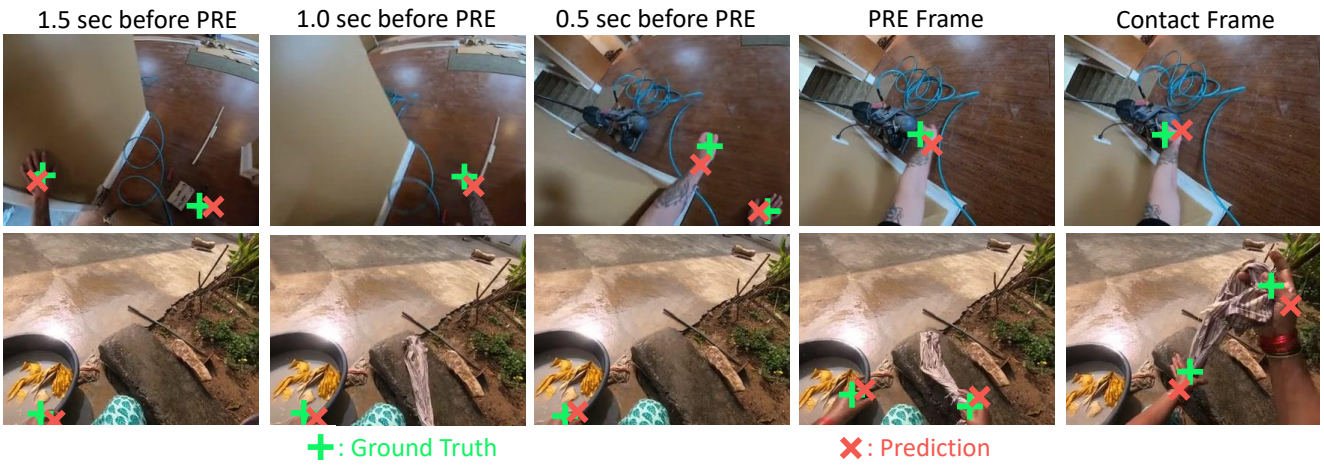


Figure 61. Qualitative examples of future hands movements prediction using the proposed baseline. The ground truth hands positions are plotted as green crosses, while the predicted hands positions are plotted as red crosses.

Figure 62. Qualitative examples of short-term object interaction anticipation using the proposed baseline. The numbers in brackets represent the confidence scores associated to the predictions. The ground truth next-active object is highlighted using a dashed red line, whereas model predictions are reported in blue solid lines.

the taxonomy where possible. In future annotations, we plan to ask annotators to always pick the closest taxonomy item even if writing in a free-form OTHER label, to encourage them to stick to the taxonomy as much as possible. Second, we noticed annotators struggled with defining bounding boxes over "stuff" categories. For example, when labeling "cutting grass", it was often challenging to draw a box that covers the full extent of the object of change (*i.e.* "grass"). Finally, it was sometimes challenging to define what the object of change was, when using large tools. For example, if using a lawn mower to clear grass, does one consider the mower as the tool and hence the grass as the object of change, or the levers and buttons inside the mower as the object of change. We chose to rely on the narrators to define which interaction to label (*i.e.* pushing the lever/button vs cutting grass), and asked the annotators to label

Figure 63. Long term action anticipation - qualitative results. Actions in green represent correct predictions (correct action, at the correct position). Actions in blue represent incorrect ordering of valid actions. Our edit-distance metric accounts for both cases.

Val set			ED@(Z=20)		
Backbone	Aggregator	Decoder	Verb	Noun	Action
SlowFast	Concat	No Change	0.766	0.830	0.960
SlowFast	Concat	MultiHead	0.747	0.808	0.952
MViT	Concat	MultiHead	**0.707**	0.901	0.972
SlowFast	Transformer	MultiHead	0.745	**0.779**	**0.941**

Test set			ED@(Z=20)		
Backbone	Aggregator	Decoder	Verb	Noun	Action
SlowFast	Concat	No Change	0.761	0.810	0.959
SlowFast	Concat	MultiHead	0.743	0.791	0.948
MViT	Concat	MultiHead	**0.697**	0.904	0.969
SlowFast	Transformer	MultiHead	0.739	**0.780**	**0.943**

Table 37. Results of the long-term action anticipation task. Lower is better. See text for discussion.

tools and objects accordingly.

Future Locomotion Movements Prediction

The baseline quantitative results on the locomotion prediction task imply that the visual cues, e.g., side walk, obstacles, and road, in egocentric images are highly indicative of future movement. However, the baseline method that encodes the visual semantics of an image with a global feature is not detailed enough to model complex walking movement, e.g., avoiding pedestrians. This opens an opportunity for challenge participants to incorporate a fine-grained visual representation.

Future Hands Movements Prediction

Our baseline model for future hands movements prediction suffers from the drastic head movements in egocentric video and the stochastic nature of future forecasting. We speculate that explicitly modeling the head movements and next-active objects may complement the video representations for predicting future hands movements.

Short-Term Object Interaction Anticipation

The short-term object interaction anticipation results highlight that the proposed task is challenging, with the baseline achieving an overall Top-5 mAP of 2.07% on the validation set and 2.45% on the test set. The key challenges are likely due to the uncertain nature of future predictions as well as to the inability of the object detector to correctly detect next active objects and ignore the others. Nevertheless, the proposed baseline, even if simple, allows to greatly improve over a combination of an object detector and a random prediction of verbs and time to contact quantities. This suggests that methods can learn to analyze the input video in order to make reasonable predictions about the future.

Long-Term Action Anticipation

We discuss several important aspects of the long-term action forecasting problem through our experiments and ablation studies. All ablations are run with SlowFast backbone networks, and models are trained for 30 epochs.

How important is Ego4D action recognition pre-training? Table 38 shows the performance of our models when pretrained *only* on Kinetics-400 action recognition (as opposed to further fine-tuning on Ego4D action recognition). All models benefit greatly from training on Ego4D data in two ways. First, there is a large domain gap between Kinetics and Ego4D both in terms of visuals (third-person vs. egocentric viewpoint) and the diversity of activities they contain, which pre-training helps account for. Second, action recognition models benefit from biases in the label structure of future actions as seen from the performance of the *No Change* baseline in Table 37.

How important is past context for transformer based models? Our transformer aggregation modules aggregate information across a larger temporal history controlled by the number of input clips to the model. Table 39 shows the sensitivity of these models to the amount of past context video that it has access to. Overall, performance increases as more context information is provided to the model, however this increase comes at the cost of memory consumption — 8 is the maximum number of clips that can be fit

Val Set			ED@(Z=20)		
Init	Backbone	Aggregator	Verb	Noun	Action
K400	SlowFast	Concat	0.752	0.820	0.958
+Ego4D	SlowFast	Concat	**0.747**	**0.808**	**0.952**
K400	SlowFast	Transformer	0.746	0.809	0.953
+Ego4D	SlowFast	Transformer	**0.745**	**0.779**	**0.941**

Table 38. Long term anticipation - varying pretraining data. MultiHead decoder used for all models. Ego4D action recognition pretraining greatly improves downstream forecasting performance.

Val Set			ED@(Z=20)		
# clips	Backbone	Aggregator	Verb	Noun	Action
2	SlowFast	Transformer	**0.743**	0.790	0.946
4	SlowFast	Transformer	0.744	0.796	0.947
8	SlowFast	Transformer	0.745	**0.779**	**0.941**

Table 39. Long term anticipation - varying number of input clips. MultiHead decoder used for all models. Performance increases with more input context.

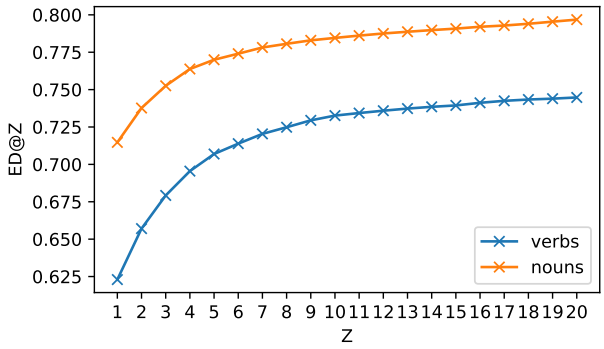


Figure 64. Performance vs. number of future actions Z. Predicting further into the future is naturally more difficult. Models begin to predict close to random actions for very high values of Z.

in GPU memory.

How far into the future can models predict? As mentioned in Section J.4 we report results for predictions at $Z = 20$ as baselines begin to predict actions at random for higher values of Z. Figure 64 shows the plot of edit distance vs. Z for our baseline models. As expected, it is far easier to anticipate actions that occur immediately next, which gets more difficult as Z increases, and steadily plateaus.

How to generate multiple candidate predictions? As mentioned in Section J.4 we evaluate the best of $K = 5$ predictions to arrive at our final results. To generate the K predictions, we sample each classifier head independently, however there are several methods to improve this including heuristic search algorithms (like beam search). Ideally, the multi-modal nature of future prediction should be accounted for in the model design itself. Moreover, decoder models that take into account the sequential nature during inference should be considered. These include transformer based decoders that are popular in recent language models (e.g., BERT, GPT) This is an important future direction of research.

J.8 Contributions statement

Giovanni Maria Farinella led the Forecasting Benchmark working on the definition of the proposed tasks, on the collection, and writing the paper.
Rohit Girdhar co-led the Forecasting Benchmark working on the definition of the proposed tasks, on the collection, and writing the paper.
Antonino Furnari contributed to the definition of the proposed benchmark tasks and in particular to the Short-Term Object Interaction Anticipation task and has been key driver of implementation, collection, annotation development throughout the project, and writing the paper.
Ilija Radosavovic worked on the definition of tasks and has been key driver of implementation, collection, annotation development throughout the project, and writing the paper.
Tushar Nagarajan contributed to the definition of the proposed benchmark tasks and in particular to the Long-Term Action Anticipation task and has been key driver of implementation, collection, annotation development throughout the project, and writing the paper.
Tullie Murrell worked on baseline implementation of the Long-Term Action Anticipation task.
Karttikeya Mangalam worked on baseline implementation, experiments and writing the Long-Term Action Anticipation task.
Christoph Feichtenhofer oversaw the development of the task, baselines and implementation of the Long-Term Action Anticipation task.
Miao Liu worked on the definition of Future Hands Movement Prediction task and has been key driver of implementation, collection, annotation development throughout the project, and writing the paper.
Wenqi Jia worked on baseline implementation of the Future Hands Movement Prediction task.
Zachary Chavis worked on the Locomotion Forecasting task and has been key driver of implementation, collection, and annotation development throughout the project.
Hyun Soo Park worked on the definition of Locomotion Forecasting tasks, collection, annotation, and writing the paper.

K. Societal Impact

Our contribution can positively impact video understanding. It offers the research community a large-scale resource captured with rigorous privacy and ethics standards (detailed in Appendix A and B) together with a diversity of subjects, and the benchmarks will promote reproducible technical advances. More broadly, egocentric perception has the potential to positively impact society in many application domains, including assistive technology, education, fitness, entertainment and gaming, eldercare, robotics, and augmented reality.

Nonetheless, future research in this area must guard against the potential negative societal impact if technology for egocentric vision were misused.

First, there are risks surrounding privacy. As we begin to see a proliferation of wearable cameras in public spaces, producers of these wearable devices will need to develop and implement protocols for notice and consent regarding the collection of data in public spaces, as well as user controls for how such data may be used, stored, and shared with any third parties. Similarly, models that may be used to transcribe speech or perform other tasks related to footage should include robust user controls such as the ability to remove or obscure personal data or sensitive content.

Note that for all our audio-visual and social benchmarking work, the data used has full consent from the participants in the video, i.e., to use their unblurred faces and audio of their conversation. To date, the research community has lacked any large-scale data resource with which to study these kinds of problems; Ego4D will help the community to consider new solutions while leveraging real-world, diverse data that respects the privacy protocols of different countries. Furthermore, the Ego4D data is available only for users who sign a license that enumerates the allowable uses of the data, which is intended to hinder potential negative applications.

Second, there is a risk that our large-scale collection could inspire future collection efforts without the same level of care or attention to the privacy and ethical concerns as were taken in Ego4D. To mitigate this risk, we have aimed to be comprehensive in our descriptions of all parts of our procedures, and we will include our best practices recommendations when publicly disseminating the results of the project.

Finally, despite our best efforts as discussed in the main paper, there are still some imbalances in the dataset. For example, the data from Rwanda is relatively small, and though 74 cities represents a leap in coverage, they do not capture all possible demographics. We acknowledge that no matter how far one goes, full global coverage of daily life activity is elusive. Still, we can mitigate this risk by continuing to grow global collaborations with researchers and participants in underrepresented areas.