Dynamic Scene Graphs for Action Recognition in Egocentric and Exocentric Videos

Mohamed Malek Abid
moabid@ethz.ch

Debmalya Chatterjee
dchatterjee@ethz.ch

Anna Rutkiewicz
annamonika.rutkiewicz@uzh.ch

Sushant Swamy
sswamy@ethz.ch

Abstract

Action recognition in dynamic and real-world environments remains a challenging task due to occlusions, complex scenes, and interactions among multiple objects. This work proposes a novel action recognition pipeline that leverages dynamic scene graphs and Temporal Graph Neural Networks (TGNNs) to model both spatial and temporal relationships in egocentric and exocentric video data. The approach begins by employing zero-shot segmentation and vision-language models (VLMs) to construct dynamic scene graphs that capture semantic and structural context across frames. These graphs serve as input to a TGNN, which learns to represent and reason over temporal interactions between entities. By integrating rich visual semantics with graph-based temporal modeling, our method aims to enhance robustness and generalization in action recognition tasks without requiring extensive manual annotations.

1. Introduction

Understanding human activities in dynamic, real-world environments is a central goal of computer vision, especially in egocentric and exocentric video analysis. Egocentric videos, in particular, offer a unique first-person perspective on how individuals interact with objects and their surroundings over long temporal spans. However, they can be very complex, with scenes characterized by occlusions, multi-object interactions, and temporal de-coherence, all of which pose significant challenges for traditional action recognition methods. [3]

Recent advancements in vision-language models (VLMs) and segmentation techniques have greatly improved our ability to extract semantically rich representations from visual data without requiring extensive manual annotation or a closed vocabulary. In parallel, graph-based representations have shown promise for memory-efficient modeling structured relationships between entities in a scene; however, they are often limited to short-range interactions and static snapshots. [7]

In this work, we propose a novel action recognition pipeline that leverages dynamic scene graphs and Temporal Graph Neural Networks (TGNNs) to model both spatial and temporal relationships across video sequences. Our approach constructs dynamic scene graphs using zero-shot segmentation, hand detection, and VLM object annotation, capturing entities and their interactions evolving over time. These graphs are then used to train a Temporal Graph Neural Network, enabling the system to learn representations over sequences of actions and interactions, which allows it to classify an arbitrary segment of a video with an action label consisting of a verb and a noun appropriate for the dataset under consideration; however, this action can also optionally be derived from an open verb/noun vocabulary.

By representing the structure and evolution of scenes as graphs, our method provides a long-term relational understanding of human activities. In doing so, our work contributes towards closing the gap between low-level visual data and high-level semantic understanding in dynamic, real-world environments, and could potentially improve current techniques used in robotics, planning, and perception.

1.1. Related Work

A recent line of work by [5] introduces EgoVideo, a model that enhances egocentric video representation learning by explicitly modeling fine-grained hand-object dynamics. Recognizing that existing video-language pre-training methods often neglect detailed physical interactions, they propose HOD, a data generation pipeline that combines hand-object detection with large language models to create rich motion-aware narrations. Their dual-branch ViT-based architecture includes a novel lightweight motion adapter designed to capture temporal cues at high frame rates. Through co-training on HOD data, EgoVideo

achieves state-of-the-art performance on multiple egocentric tasks. While our approach similarly focuses on dynamic scene understanding, we aim to build more interpretable models of the higher-level hand-object relations with dynamic scene graphs, and then perform the classification using trained Temporal Graph Neural Networks. Nonetheless, the success of these E2E methods such as EgoVideo and [9] indicates the value of structured motion-aware representations for egocentric video understanding.

Recent efforts in egocentric video understanding have begun emphasizing hand-object interactions as key cues for activity recognition and representation learning. While prior work often treated hands and objects as separate semantic entities or used hand masks as auxiliary inputs, these approaches typically failed to capture the nuanced, contact-based relationships critical to egocentric interaction. Addressing this, CaRe-Ego [10] introduces a contact-aware segmentation framework for the EgoIHOS task, explicitly modeling both hand-object and object-object relationships. The proposed Hand-guided Object Feature Enhancer (HOFE) leverages hand features to enhance object representations through cross-attention, while the Contact-centric Object Decoupling Strategy (CODS) disentangles overlapping object classes (e.g., two-hand objects) by focusing on interaction patterns instead of rigid classification. This dual-focus on interaction and contact not only reduces semantic ambiguity but also improves segmentation generalization across domains. There have also been extensions to state-of-the-art video understanding approaches using TGNNs, for instance TESGNN [6], which uses equivariance constraints on the graph for robust scene understanding. Finally, the foundation of our investigation relied on two datasets: EPIC Kitchens-100 [11] and Ego-Exo4d [3], both of which contain egocentric video segments with labels corresponding to common tasks in the kitchen and household.

2. Methodology

2.1. Overview

We propose an approach that effectively combines several state-of-the-art models for segmentation, point tracking, hand-object interactions, etc. We begin by decomposing video frames into their constituent objects using the Segment Anything Model (SAM) [8]. Due to storage capacity and computational (GPU) constraints, we restrict ourselves to operating on Egocentric video from the Epic-Kitchens dataset, which simplifies our person-object relation detection method, since we can specifically focus on identifying hands as the primary agents of action using a dedicated detector. Knowing what both hands are interacting with is a strong foundation for human action recognition, especially in an egocentric setting. However, our approach should theoretically be easily extendable to the Ego4D egocentric data; all the required additional pre-processing in the form of undistorting the recording using camera intrinsics has been integrated into our pipeline and is functional albeit computationally expensive to run on a minimally viable subset of data. However, for exocentric data, a more extensible body-part detection system (skeletal/pose estimation) would be necessary to support actions not commonly performed using hands, such as kicking a soccer ball.

One of the strengths of our methods arises from our representation of a scene, which we construct by taking segmented objects and detected right/left hands, then leveraging the open-vocabulary world knowledge of a Vision Language Model (VLM) to descriptively label relations between objects and any visible hands in the frame. Two objects relating to one another (e.g., "right hand *holding* knife"), would be encoded as a semantic edge in our graph. Simultaneously, we capture the scene's temporal dynamics using a tracking model (CoTracker [4]) to follow objects across frames, establishing the temporal links that define the action's progression. Thus, after constructing our structured graph, we have encoded both "what" is happening and "how/when" it evolves; finally, these dynamic scene graphs are then used to train a Temporal Graph Neural Network (TGN) for robust and context-aware action classification.

We define the task in Sec. 2.2, followed by our three-stage pipeline: (1) Per-Frame Scene Pre-processing (Sec. 2.4), (2) Dynamic Scene Graph Construction (Sec. 2.5), and (3) Temporal Graph-based Action Classification (Sec. 2.6). Training and inference are detailed in Sec. 3.

2.2. Task Formulation

Our goal is to classify actions in untrimmed egocentric videos. Each input is a video segment with metadata (e.g., participant ID, narration ID), and the output is an action label (`verb + noun`, e.g., "take bottle"). We construct a dynamic scene graph representing object interactions over time, then classify actions using a Temporal Graph Neural Network (TGNN).

2.3. Pipeline Overview

Figure 1 shows the three-stage pipeline:

1. **Per-Frame Scene Pre-processing (Offline)**
   Processes each video frame using SAM2 to extract instance masks. Saves in HDF5 for efficiency.
2. **Dynamic Scene Graph Construction**
   Detects hands and interacting objects; tracks object instances using CoTracker; uses an LLM-based descriptor for semantic labels and relationships. Outputs a structured JSON scene graph.

3. **Temporal Graph-based Action Classification**
   Uses CLIP to embed node labels. A heterogeneous GNN with spatial and temporal edges predicts the action label.

2.4. Stage 1: Per-Frame Scene Pre-processing

2.4.1 Instance Segmentation (SAM2)

- Extract frames (e.g., every 8th) to reduce redundancy.
- Use SAM2 to segment all objects; store binary masks in HDF5.

2.4.2 Hand and Interaction Detection

- Use `hands23` or `mediapipe` to detect hands and objects they interact with.
- Filter SAM2 masks to retain only interacting objects (via overlap or direct output).
- Draw red ellipses for VLM-based labeling.

2.5. Stage 2: Dynamic Scene Graph Construction

2.5.1 Initialization (First Frame)

- Create `Object` nodes from masks.
- Use an VLM descriptor for semantic labels and relations between hands/objects
- Create `Hand` nodes and relational edges.
- Sample N points per mask for tracking.

2.5.2 Temporal Linking (CoTracker)

- Track points across frames: (t, x, y).
- CoTracker predicts new locations and visibility scores.

2.5.3 Graph Updating

- Match tracked points to new masks:
  - **Match:** Update node.
  - **No Match:** Create new node, label via LLM, sample points.
  - **Object Lost:** Mark node as invisible.
- Final graph (nodes, edges, labels, timestamps) saved as JSON.

2.6. Stage 3: Temporal Graph-based Action Classification

2.6.1 Model Architecture

- **Node Embedding:** Use CLIP (512D) to encode LLM-generated labels.
- **Heterogeneous GNN:**
  - `GATConv` for spatial edges (object-object).
  - `TransformerConv` for temporal edges (same object across frames).
- **Pooling and Classification:** Mean-pool node features $\rightarrow$ 2-layer MLP $\rightarrow$ action logits.

Algorithm 1 Dynamic Scene Graph Construction

Input: Video V, Hand Detector D_H, LLM D_{LLM}, Tracker T_p
Output: Scene Graph G
Init G
for each chunk in V **do**
  Extract frame F
  Detect hands and interacting objects
  Filter masks $\{M_j\}$ from SAM2
  Predict point locations $\{p_i'\}$ via T_p
  Match $p_i' \rightarrow M_j$
  for each match (O_i, M_j) **do**
    Update O_i with M_j
  end for
  for each unmatched M_k **do**
    Create node O_k, label via D_{LLM}
    Add edges, sample points, update T_p
  end for
  for unmatched objects **do**
    Mark as invisible if obj unmatched for $\geq frame\_unmatched\_limit$
  end for
end for
return G

3. Experiments

3.1. Datasets

We use the Epic Kitchens[2] dataset for our experiments. Specifically, we use an open sourced version of the dataset provided in [12] where the videos are cut into 15 second long chunks and are resized to a smaller size.

Preprocessing First, we extract the 10 most frequently occurring noun and verb classes from the dataset, and filter actions based on these classes.

Dataset size The training set consisted of 836 randomly sampled segments, and the validation set consisted of 200 segments.

3.2. Metric

In order to evaluate our method's performance on action detection/recognition tasks featured as workshop challenges for CVPR 2025, we implemented the **mean Average Precision** metric, following the EK100 Action Detection challenge [11]. We also used the baseline metrics provided in the EK100 Action Recognition challenge, which were *top acc. @ 1*, *top acc. @ 5*, for actions, verbs, & nouns.

Let (C) be the set of all classes. First, for each class, the temporal intersection over union is calculated as follows:

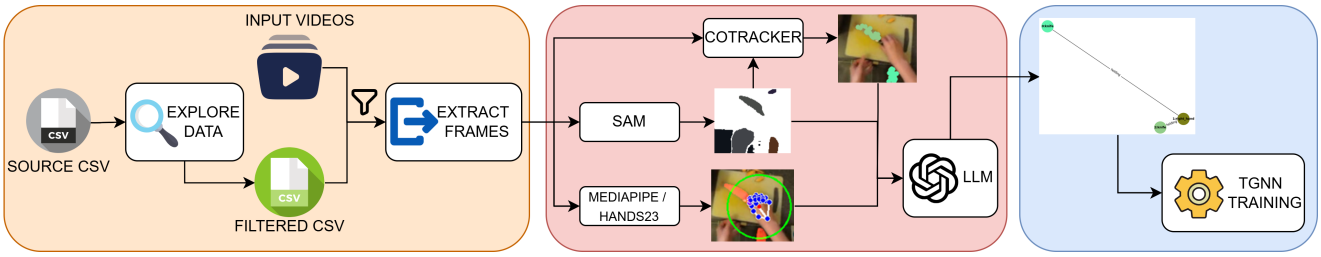


Figure 1. Three-stage pipeline: frame extraction, dynamic scene graph construction, and TGNN-based action classification.

tIoU$(S_p, S_{gt}) = \frac{|S_p \cap S_{gt}|}{|S_p \cup S_{gt}|}$, where S_p - predicted segment, S_{gt} - ground truth segment. Based on that, we classify the predictions either as True Positive (highest-scoring prediction matched to an unassigned ground truth segment with tIoU$\geq$ threshold) - or False Positive (otherwise).

We compute precision and recall for all sorted predictions. Next, the interpolated average precision (AP_c) for class c is calculated: $\text{AP}_c = \sum_k (r_k - r_{k-1}) p_{\text{interp}}(r_k)$, where $p(r)$ is the precision as a function of recall, and $p_{\text{interp}}(r) = \max_{r' \geq r} p(r')$. The **mAP** is then defined as the mean of the AP scores calculated across all action classes and all thresholds: $\text{mAP} = \frac{1}{|C||T|} \sum_{c \in C} \sum_{t \in T} \text{AP}_c(t)$, where C - set of all classes, T - set of all thresholds.

(a) hands23 w/ output (b) mediapipe

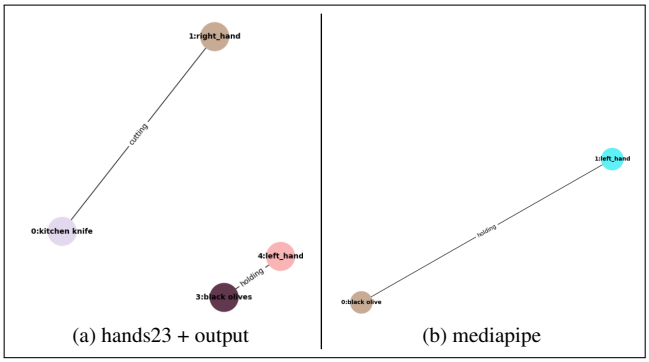
Figure 2. Hand detector comparison.

3.3. Hand/Object Detection and Scene Graphs

Parameter Finetuning We tested out different hyperparameter values, and, after an analysis of the results, we decided to make the following parameter changes:

- **segmentation**: decrease `min_px` from 500 to 200,
- **mediapipe**: decrease `radius_around_hands` from 70 to 50, decrease `landmarks_th` from 0.3 to 0.15
- **chatgpt**: change the model from `gpt-4o` to `gpt-4.1`, increase the `temperature` from 0 to 0.5, change `top_p` from 0.2 to default.

We also added and specified the config parameters for `hands23`.



(a) hands23 + output (b) mediapipe

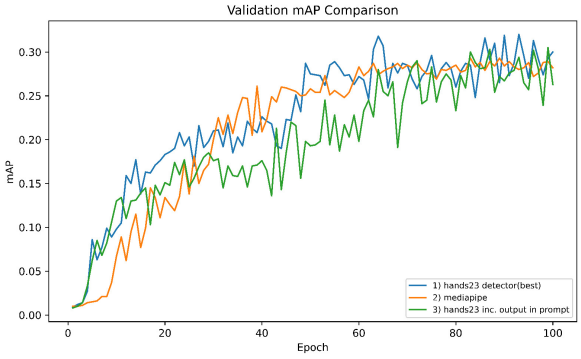
Figure 3. Scene graph comparison.

VLM Comparison We performed scene graph labeling both using GPT and Claude. We then visually compared the resulting graphs and concluded that GPT provides better results.

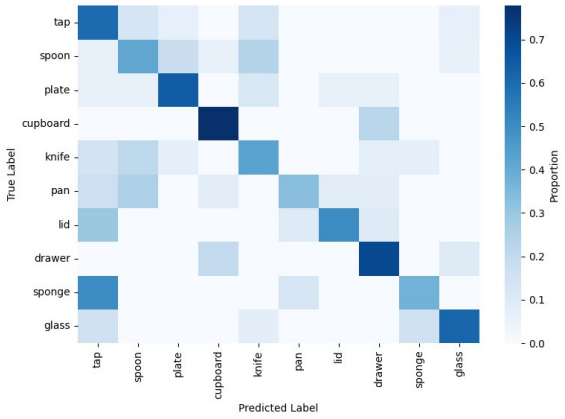
Prompt Engineering We also crafted and tested a set of VLM prompts, visually compared the VLM outputs, and selected the prompt achieving the best results, which we then used for further evaluations. Initially, the prompt that our inherited pipeline had out-of-the-box yielded limitations in the form of highly repetitive verb labels: for instance, the VLM would often exclusively use the terms "grasping" or "holding" to label any hand-object interaction.

We first tuned temperature, top_p, and then modified the prompt with a richer context in regards to descriptive verbs and nouns, ensuring that we weren't explicitly giving away any important tips or instilling any biases (e.g telling the VLM that it is only to use certain words for kitchen activities). Furthermore, we added an additional image to the prompt: a frame with all points that were being tracked highlighted, which gave the model a bit of extra specificity when choosing which noun to focus on for relation annotations. Finally, we noticed that the VLM was limited by only being able to return one annotation per rame, so we changed the prompt to elicit a ranked (and confidence-scored) list of the most probable hand-object interactions in a given frame, which also improved our qualitative results once we established heuristics for selecting or discarding annotations.

Hand Detector Comparison We also performed a visual comparison of the scene graphs achieved using both



(a) mAP metric score in relation to the number of epochs.



(b) Noun confusion matrix for the best performing noun-accuracy-wise method - `hands23`.

Figure 4. TGNN training results on the validation dataset.

hand detectors and unanimously decided that `hands23` performs better in this qualitative analysis. Sample comparison can be found in Figures 2 and 3. We can clearly see that `hands23` manages to correctly detect both hands and interacting objects, while `mediapipe` output is limited to the left hand only.

We also evaluated the number of graphs created based on outputs of each of the detectors - the results can be found in Table 3. Again, `hands23` performs significantly better, as its detections result in our pipeline creating up to 50% more graphs compared to `mediapipe`.

3.4. Action Recognition

We have evaluated and compared two hand detectors: `mediapipe` and `hands23` on the EK100 dataset. However, the `hands23` detector offers additional information that we did not use in our initial experiment, so we performed an extension to our work. As `hands23`'s method outputs a JSON dictionary that has specific information about each hand's contact state and grasp type, we were thus able to augment the VLM's context with highly-accurate information about the hand contact states and type of grasp. This auxiliary information helped steer the actions that the VLM would predict to a better subspace, especially when it came to verbs.

An example of this would be using the grasp classification of the Power-Circular or Prismatic [1] to classify something as "holding" or even "cutting" instead of "touching". We compared the resulting scene graphs from all three methods as well as presented the results on the training and validation sets (800/200 samples from EK-100) in Table 2. We can observe that `mediapipe` is probably overfitting, as it performs best out of the three methods on the training dataset, and worst on the validation dataset. This is most likely due to the fact that it is trained on only 425 scene graphs, compared to 637 or 675 scene graphs on which `hands23` is trained, thus certain noun/verb combinations are more likely to fall on the tail of the learned distribution for classification.

Figure 4 presents training results of our method on the validation dataset. We can clearly see that mAP score plateaus around epoch 80. Furthermore, the noun confusion matrix looks reasonably well, with only a couple of frequent mismatches - such as *sponge* and *tap*, which are an understandable mistake that may stem from the fact that these objects often occlude or occur next to each other in egocentric video, especially in the kitchen-domain.

Method	Acc@1	Acc@5	V. Acc.	N. Acc
mediapipe [T]	**0.875**	**1.000**	**0.903**	**0.894**
hands23 [T]	0.781	0.996	0.828	0.856
hands23 + o [T]	0.761	0.998	0.808	0.852
mediapipe [V]	0.362	0.550	0.475	0.512
hands23 [V]	**0.408**	0.664	0.464	**0.536**
hands23 + o [V]	0.382	**0.687**	**0.481**	0.511

Table 1. Results on the training [T] and validation [V] sets for `mediapipe`, `hands23` and `hands23` with its output passed to the LLM. mAP is the mean average precision as in Subsec. 3.2, Acc@1 and Acc@5 are top-1 and top-5 accuracies respectively, V. Acc. is the verb accuracy (accuracy calculated when focusing only on the verb part of the label), and N. Acc is the noun accuracy (accuracy calculated when focusing only on the noun part of the label).

4. Contributions

Domain contributions We proposed a novel pipeline for action recognition that leverages scene graphs and

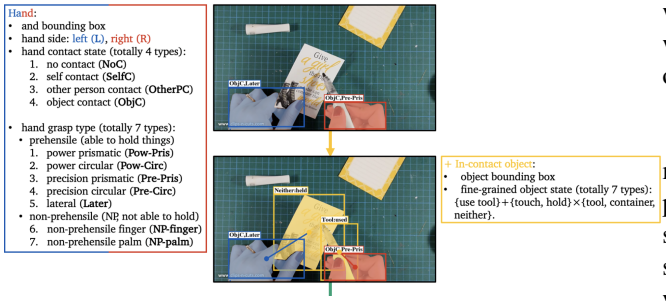
Approach	**mAP**
mediapipe	0.293
hands23	**0.310**
hands23 + o	0.305
benchmark*	0.3197

Table 2. Results of our method compared to EK100 Action Detection challenge winners.
*Due to storage and compute limitations, our method was trained and evaluated on a significantly smaller subset of EK100 (0.8/0.2 split on 1000 samples from the full-sized ∼77,000 sample dataset) as compared to SOTA methods from the CVPR 2025 challenge.

Hand Detector	Graphs Created
mediapipe	425
hands23	637
hands23 + o	**675**

Table 3. Comparison of the three object detectors in terms of how many frames did the hand detector construct a scene graph. All hand detectors were run on the same dataset.

Temporal Graph Neural Networks (TGNNs). As part of this pipeline, we enriched the vision-language model (VLM) scene labeling prompt using grasp information extracted from the **Hands23** output, leading to a notable improvement in verb prediction accuracy. To the best of our knowledge, such integration of TGNNs and scene graph-based reasoning has not been extensively explored in the literature of our domain.



4.1. Work distribution

Anna Contributed to automating the EPIC-Kitchens pipeline. Improved scene graph labeling via prompt engineering and performed the VLM comparison. Integrated the mAP metric into the pipeline.

Debmalya Helped to tune parameters in the pipeline, and automate the EK100 data preprocessing, and fixing bugs in the Ego4D pipeline. Helped run experiments and optimizing the segmentation code for EK100 pipeline.

Malek Contributed to parameter tuning for segmentation/object tracking. Automated Ego4D pipeline, assisted with EK100 data-preprocessing and evaluated small subset of EgoExo4D for viability. Ran experiments and surveyed prior work for potential inspiration. Contributed to prompt engineering and Hands23 grasp-aware integration.

Sushant Contributed to data preprocessing and prepared the action recognition pipeline for the EPIC-Kitchens dataset. Additionally, integrated the Hands23 model into the pipeline to enhance grasp-aware scene understanding.

4.2. Code Contributions

The initial code for the pipeline was provided by our supervisors (GitHub Link). Code contributions include rewriting the pipeline to use a variable/scalable amount of data from the EPIC Kitchens dataset and tuning the config parameters in the original pipeline. We also integrate the `hands23` detection model into the pipeline by adapting the code provided (GitHub Link). We also add the mAP metric in the training script to evaluate the models. Our final code can be found here (GitHub Link) which is also included in the submission.

5. Conclusion

Motivated by the sizable scope and richness of Egocentric and Exocentric data, we explored recent datasets such as EgoExo4D, as well as older benchmarks in the field that remain formidable (the EK-100 CVPR 2025 Action Recognition & Detection challenges). While we were limited by computational constraints that made working with the large files from Meta's Aria glasses difficult, we were still able to reach competitive performance on a subset of the EK-100 dataset.

Fundamentally, our approach stands out from many modern attempts to recognize actions from video, most importantly due to the interpretable and efficient graph representation that we exploit in this work. By encoding both spatial and temporal relations into dynamic scene graphs, we were able to effectively train our classifier to exploit these representations for the downstream tasks we chose to explore. Furthermore, we benchmarked on various sample sizes as we worked our way up to $\approx$ 1000 videos, and noticed a consistent trend where training accuracy decreased and validation accuracy increased as we added more data. Thus, we hypothesize that our approach would continue to scale without any collapse, especially when trained on all available data from EK-100. Improving the Temporal Graph Network and the EgoExo4D segmentation and processing pipeline seem to be promising avenues for future improvement in this domain.

References

[1] Tianyi Cheng, Dandan Shan, Ayda Sultan Hassen, Richard Ely Locke Higgins, and David Fouhey. Towards a richer 2d understanding of hands at scale. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. 5

[2] Dima Damen, Hazel Doughty, Giovanni Maria Farinella, , Antonino Furnari, Jian Ma, Evangelos Kazakos, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, and Michael Wray. Rescaling egocentric vision: Collection, pipeline and challenges for epic-kitchens-100. *International Journal of Computer Vision (IJCV)*, 2021. 3

[3] Kristen Grauman, Andrew Westbury, Lorenzo Torresani, Kris Kitani, Jitendra Malik, Triantafyllos Afouras, Kumar Ashutosh, Vijay Baiyya, Siddhant Bansal, Bikram Boote, Eugene Byrne, Zach Chavis, Joya Chen, Feng Cheng, Fu-Jen Chu, Sean Crane, Avijit Dasgupta, Jing Dong, Maria Escobar, Cristhian Forigua, Abrham Gebreselasie, Sanjay Haresh, Jing Huang, Md Mohaiminul Islam, Suyog Jain, Rawal Khirodkar, Devansh Kukreja, Kevin J Liang, Jia-Wei Liu, Sagnik Majumder, Yongsen Mao, Miguel Martin, Effrosyni Mavroudi, Tushar Nagarajan, Francesco Ragusa, Santhosh Kumar Ramakrishnan, Luigi Seminara, Arjun Somayazulu, Yale Song, Shan Su, Zihui Xue, Edward Zhang, Jinxu Zhang, Angela Castillo, Changan Chen, Xinzhu Fu, Ryosuke Furuta, Cristina Gonzalez, Prince Gupta, Jiabo Hu, Yifei Huang, Yiming Huang, Weslie Khoo, Anush Kumar, Robert Kuo, Sach Lakhavani, Miao Liu, Mi Luo, Zhengyi Luo, Brighid Meredith, Austin Miller, Oluwatumininu Oguntola, Xiaqing Pan, Penny Peng, Shraman Pramanick, Merey Ramazanova, Fiona Ryan, Wei Shan, Kiran Somasundaram, Chenan Song, Audrey Southerland, Masatoshi Tateno, Huiyu Wang, Yuchen Wang, Takuma Yagi, Mingfei Yan, Xitong Yang, Zecheng Yu, Shengxin Cindy Zha, Chen Zhao, Ziwei Zhao, Zhifan Zhu, Jeff Zhuo, Pablo Arbelaez, Gedas Bertasius, David Crandall, Dima Damen, Jakob Engel, Giovanni Maria Farinella, Antonino Furnari, Bernard Ghanem, Judy Hoffman, C. V. Jawahar, Richard Newcombe, Hyun Soo Park, James M. Rehg, Yoichi Sato, Manolis Savva, Jianbo Shi, Mike Zheng Shou, and Michael Wray. Ego-exo4d: Understanding skilled human activity from first- and third-person perspectives, 2024. 1, 2

[4] Nikita Karaev, Ignacio Rocco, Benjamin Graham, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. CoTracker: It is better to track together. 2023. 2

[5] Baoqi Pei, Guo Chen, Jilan Xu, Yuping He, Yicheng Liu, Kanghua Pan, Yifei Huang, Yali Wang, Tong Lu, Limin Wang, and Yu Qiao. Egovideo: Exploring egocentric foundation model and downstream adaptation, 2024. 1

[6] Quang P. M. Pham, Khoi T. N. Nguyen, Lan C. Ngo, Truong Do, Dezhen Song, and Truong-Son Hy. Tesgnn: Temporal equivariant scene graph neural networks for efficient and robust multi-view 3d scene understanding, 2025. 2

[7] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision, 2021. 1

[8] Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, Eric Mintun, Junting Pan, Kalyan Vasudev Alwala, Nicolas Carion, Chao-Yuan Wu, Ross Girshick, Piotr Dollár, and Christoph Feichtenhofer. Sam 2: Segment anything in images and videos, 2024. 2

[9] Tsukasa Shiota, Motohiro Takagi, Kaori Kumagai, Hitoshi Seshimo, and Yushi Aono. Egocentric Action Recognition by Capturing Hand-Object Contact and Object State . In *2024 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 6527–6537, Los Alamitos, CA, USA, 2024. IEEE Computer Society. 2

[10] Yuejiao Su, Yi Wang, and Lap-Pui Chau. Care-ego: Contact-aware relationship modeling for egocentric interactive hand-object segmentation, 2025. 2

[11] EPIC-KITCHENS Team. C2-Action-Detection: Official github repository for the epic-kitchens c2 action detection challenge. https://github.com/epic-kitchens/C2-Action-Detection, 2025. Accessed: 2025-06-23. 2, 3

[12] Yue Zhao and Philipp Krähenbühl. Training a large video model on a single machine in a day. *arXiv preprint arXiv:2309.16669*, 2023. 3