HiERO: understanding the hierarchy of human behavior enhances reasoning on egocentric videos

HiERO: understanding the hierarchy of human behavior enhances reasoning on egocentric videos

Simone Alberto Peirone
Politecnico di Torino
simone.peirone@polito.it

Francesca Pistilli
Politecnico di Torino
francesca.pistilli@polito.it

Giuseppe Averta
Politecnico di Torino
giuseppe.averta@polito.it

Abstract

Human activities are particularly complex and variable, and this makes challenging for deep learning models to reason about them. However, we note that such variability does have an underlying structure, composed of a hierarchy of patterns of related actions. We argue that such structure can emerge naturally from unscripted videos of human activities, and can be leveraged to better reason about their content. We present HiERO, a weakly-supervised method to enrich video segments features with the corresponding hierarchical activity threads. By aligning video clips with their narrated descriptions, HiERO infers contextual, semantic and temporal reasoning with an hierarchical architecture. We prove the potential of our enriched features with multiple video-text alignment benchmarks (EgoMCQ, EgoNLQ) with minimal additional training, and in zero-shot for procedure learning tasks (EgoProceL and Ego4D Goal-Step). Notably, HiERO achieves state-of-the-art performance in all the benchmarks, and for procedure learning tasks it outperforms fully-supervised methods by a large margin (+12.5% F1 on EgoProceL) in zero shot. Our results prove the relevance of using knowledge of the hierarchy of human activities for multiple reasoning tasks in egocentric vision. Project page: sapeirone.github.io/HiERO.

1. Introduction

Think about a typical home routine. You enter in the kitchen and grab onions and carrots, chop them, and put them in a pan on the stove with oil. At the same time, you fill a pot with water and put it on the stove. While you wait for the water to boil to cook the pasta, you pour some tomatoes in the pan. Zooming out a bit, you can group all these actions into higher-level interleaved activity threads, such as *preparing vegetables* and *cooking pasta*. Looking at the bigger picture, both of these threads are part of a broader routine like *preparing a meal*, which may overlap with others, such as *washing the dishes*. Foundational models in egocentric video understanding have long focused mostly on action-level understanding [2, 15, 38, 49, 56], overlooking the inherent hierarchical nature behind human actions [7, 9, 47]. The closest class of approaches that attempts to learn about this compositional structure is Procedure Learning (PL), which assumes that multiple actions concur to form key-steps of long-horizon procedures. However, supervised approaches consider only one level of aggregation, i.e. actions that form key-steps, and require multiple scripted examples of the same procedures to learn from. Conversely, we claim that there is significant value in learning from the hierarchy of human behavior at multiple levels of abstraction. Indeed, the richness of human activities lies not only in single actions execution, but more prominently in how these are interconnected at different levels of abstractions. Our intuition is that enriching action features with knowledge of the multiple progressive semantic aggregations they belong to can significantly improve their expressiveness for various reasoning tasks. Interestingly, we believe that such a hierarchical structure can naturally emerge without specific supervision. Previous works have shown that even a simple clustering of video segments projected in feature space may be sufficient to identify the high-level activities represented in the video [4, 8, 39, 43]. However, the choice of the features extractor plays a crucial

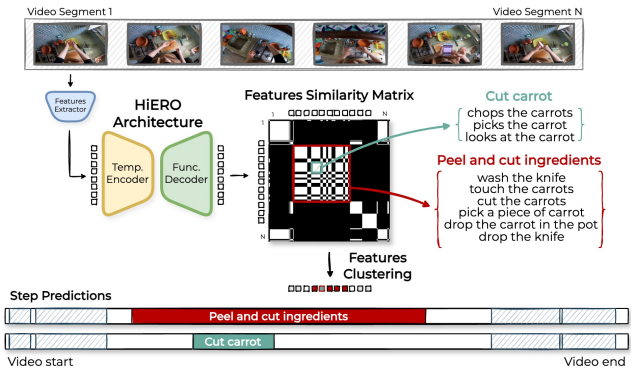


Figure 1. **Zero-Shot procedure step localization with HiERO**. Given a long egocentric video, HiERO computes segment-level features that encode the functional dependencies between the actions in the video at different scales. This enables the detection of procedure steps through a simple clustering in feature space.

role in terms of the abstractions that the clustering is able to capture. Static biases of video models [22] can generate clusters based on *visual similarity* of the video segments, for example, when two actions occur in the same environment [34]. Likewise, *semantic similarities* emerge when grouping action segments with similar semantics, *e.g.*, *dicing a carrot* and *slicing an onion*. Video-language alignment between short video clips and their corresponding textual descriptions, as seen in EgoVLP [28], results in clip-level representations that capture these similarities by leveraging the proximity of actions in the text space. At a higher abstraction level, **functional similarity** groups segments based on their functional objectives, such as identifying all the steps needed to *prepare a meal*.

With this work, we demonstrate that such functional patterns can emerge naturally from data at different abstraction scales, and can be exploited to enrich features used to solve multiple video understanding tasks in zero shot. We model the task as a graph learning problem and represent videos as graphs where nodes correspond to fixed-length video segments and edges reflect the temporal distance between nodes. To preserve and exploit the intrinsic hierarchy of human behavior, we propose to use a hierarchical graph-based representation that provides a strong inductive bias. This is implemented through a hierarchical architecture inspired by Graph U-Net [14] which we call HiERO. The model consists of a Temporal Encoder, which gradually aggregates information from nearby nodes within local temporal neighborhoods, and a Function-Aware Decoder which is responsible for discovering strongly connected regions via spectral graph clustering and performs temporal reasoning within each partition separately. In this context, activity patterns emerge as strongly connected regions capturing actions that are functionally and temporally related, allowing the model to reason on higher-level activities, see Fig 1. HiERO is trained in a weakly-supervised manner, with the objectives of aligning node features at higher temporal granularity by leveraging video-narration alignment within a temporal window, and guiding clustering at deeper layers to enforce intra-cluster feature proximity. HiERO can perform a wide set of reasoning tasks, including natural language queries, procedure learning, step grounding, and others. We evaluate the zero-shot transfer of HiERO over EgoProceL [4] and Ego4D Goal-Step [47], demonstrating remarkable performance compared to fully-supervised models, despite no explicit task-specific training. By leveraging the inner hierarchical structure of videos, HiERO is competitive also with state-of-the-art models on video-text alignment benchmarks with minimal additional training.

2. Related works

Long-form understanding. Long-form video understanding in egocentric vision requires diverse reasoning abilities to grasp the broader context of human activities [19, 21, 30, 36], interpret interactions between objects, people, and locations [16, 33, 34, 39], and model the procedural nature of human activities [3, 44, 45]. Several approaches learn transferable representations for downstream video understanding tasks by aligning short video clips and their corresponding textual narrations [2, 28, 38, 56]. HierVL [2] extends this approach by incorporating video-level alignment through summaries. Most closely related to ours, `Paprika` [58] exploits supervision from Procedural Knowledge Graphs sourced from wikiHow to develop a set of procedure-aware pre-training objectives. ProcedureVRL [57] learns procedure step representations via video-language alignment and a probabilistic model to encode temporal dependencies between individual steps in instructional videos. Unlike these approaches, HiERO captures long-range functional dependencies between human actions without requiring explicit supervision or instructional video datasets.

Procedure learning. PL involves identifying key-steps, *i.e.*, the actions required to complete a task, and predicting their logical order in videos after observing multiple visual demonstrations. Supervised approaches [35, 59] rely on per-frame key-step annotations across videos, while weakly supervised methods [26, 41, 60] leverage predefined key-step lists [1, 29, 31, 58]. These approaches require extensive annotation efforts, full video observations, or heuristic definitions, making them challenging to scale [12]. To mitigate these limitations, self-supervised methods [4, 5, 8, 11] have gained attention, as they avoid the need of per-frame annotations. These methods exploit the structured nature of multiple demonstrations of the same task to discover and localize key steps. However, they still rely on the assumption that corresponding actions exist across videos, requiring datasets that contain multiple instances of the same procedure with a shared set of key-steps for alignment. This assumption significantly limits their applicability to real-world, unscripted human activity datasets, restricting their use to well-defined procedural tasks. In contrast, HiERO effectively uncovers meaningful functional threads from unscripted videos without relying on explicit supervision.

Clustering for vision applications. Clustering approaches have been explored to localize objects in the image by looking at densely connected regions of the image [32, 46, 52, 53]. Self-supervised methods in Procedure Learning [4, 5, 8] use clustering algorithms to identify procedure steps from features extracted by a self-supervised network trained to align steps across multiple videos of the same task. TW-FINCH [43] tackles unsupervised action segmentation through hierarchical clustering of video segments in feature space, showing that clustering algorithms are surprisingly strong baselines for action segmentation. Similarly, Kumar *et al.* [25] leverages frames clustering as a pretext task, enforcing order-preserving constraints on

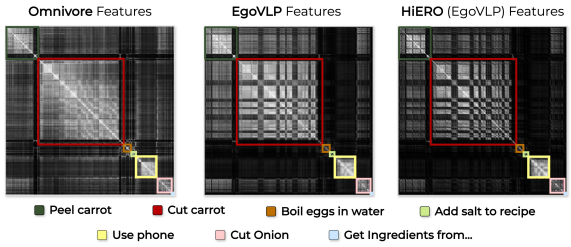


Figure 2. **Emergence of step clusters in the features similarity matrix of a video from Ego4D [17].** Colored rectangles indicate the ground truth steps. Ideally, we expect high similarity (brighter regions) if two segments represent the same or semantically similar steps, *e.g. cut onion* and *cut carrot*. On Omnivore features, this behavior is only partially visible. On EgoVLP features, we observe sharper clusters of temporal segments that are not necessarily close temporally, but represent similar high-level actions. Our approach makes this behavior even more visible.

cluster assignments across videos. These works are designed for datasets with repetitive and isolated tasks, *e.g.*, 50-Salads [48] and Breakfast [23]. Differently, HiERO captures more general functional dependencies between human actions from *in-the-wild* videos, without the need for procedural videos during training.

3. Method

We design HiERO based on the intuition that, given a sufficiently large collection of videos capturing human activities in-the-wild, *functional dependencies* between actions naturally emerge as frequently co-occurring patterns directly from observations [44]. With HiERO, we learn a feature space that captures these functional dependencies between actions, *i.e.*, those that frequently co-occur together are close to each other and distant from the others. As a result, such space allows related actions to be easily grouped into high-level patterns with a simple clustering operation. Our approach represents the video as a graph, in which nodes correspond to short temporal segments, ideally representing one or a few actions, and detects functional threads as regions of this graph whose nodes encode similar actions based on their feature similarity. Our method builds on spectral graph theory to identify these strongly connected regions of the graph (Sec 3.1), learns to detect functional threads by leveraging the natural co-occurrence of human actions in unscripted videos (Sec. 3.2) with the goal of performing different video understanding tasks without additional training (Sec. 3.3).

Functional threads discovery by graph clustering. In our context, we define a strongly connected region of the graph as a subset of its nodes showing high *functional similarity*. The concept of *similarity* is strongly dependent on the backbone used to compute the node embeddings. If the backbone was able to map close in feature space the video segments encoding similar actions, *e.g.*, *cutting an onion* and *peeling a carrot*, then these region with high similarity would correspond to high-level *functional threads*, *e.g.*, *preparing the vegetable*. To support this intuition, we show in Fig. 2 the impact of different backbones on the features similarity matrix. Omnivore was trained for supervised image and action classification on image and video data respectively. As a result, it focuses mostly on visual similarity between the segments. On EgoVLP features, some strongly connected regions emerge more clearly, even though the model was trained only with fine-grained narrations supervision. In the context of procedural videos, these regions may correspond to different steps and substeps of the procedure. Our approach builds on this intuition to make the unsupervised clustering into high-level functional threads more evident. The final goal is to partition an input graph $\mathcal{G}$ into a set of n sub-graphs $\{\mathcal{G}_1, \dots, \mathcal{G}_n\}$, each encoding a different step from the input video.

3.1. Background: Graph Theory

Let $\mathcal{G}$ be an undirected graph with node embeddings $\mathbf{X} \in \mathbb{R}^{N \times D}$, where N is the number of nodes and D the embedding size. The weighted adjacency matrix $\mathbf{W} \in \mathbb{R}^{N \times N}$ is a nonnegative matrix whose entry w_{ij} is the weight of the edge between nodes i and j, while the degree matrix $\mathbf{D} \in \mathbb{R}^{N \times N}$ is a diagonal matrix where $D_{ii} = \sum_j \mathbf{W}_{ij}$ is the degree of node i. The Laplacian matrix $\mathbf{L} = \mathbf{D} - \mathbf{W}$ of the graph is a real symmetric matrix that describes how information flows on the graph: $(\mathbf{LX})_i = \sum_{j=1}^{n} w_{ij}(\mathbf{x}_i - \mathbf{x}_j)$. The spectral decomposition of the Laplacian Matrix can reveal important topological properties of the graph. Most notably, its smallest eigenvalue λ_1 is zero and the corresponding eigenspace is formed by a set of indicator vectors that identify the connected components of the graph [50].

Graph clustering. Spectral clustering [50] groups nodes of the graph such that nodes in each partition are similar to each other. Unlike other clustering approaches, *e.g.*, K-Means, which require specific assumptions about the data distribution, spectral clustering looks at the connectivity of the graph to groups nodes. Given a target number of clusters K to separate, nodes are first projected on the subspace spanned by the eigenvectors of the normalized Laplacian matrix corresponding to its K smallest eigenvalues [50]. Then, K-Means is used to cluster the nodes in this subspace. Given the node embeddings $\mathbf{X}$ of the graph, we build the corresponding similarity matrix $\mathbf{S} \in \mathbb{R}^{N \times N}$ as $S_{ij} = \exp\left(\mathbf{x}_i^T \mathbf{x}_j / (\kappa ||\mathbf{x}_i||_2 ||\mathbf{x}_j||_2)\right)$, where κ is a temperature parameter. We define a fully connected *similarity graph* $\mathcal{G}_S$ using $\mathbf{S}$ as adjacency matrix, and define the corresponding normalized Laplacian matrix as $\tilde{\mathbf{L}}_S = \mathbf{I} - \mathbf{D}^{-\frac{1}{2}}\mathbf{S}\mathbf{D}^{-\frac{1}{2}}$, where $\mathbf{I}$ is the identity matrix and $\mathbf{D}$ is the degree matrix of $\mathcal{G}_S$. Then, we find the eigendecomposition of $\tilde{\mathbf{L}}_S$ as $\tilde{\mathbf{L}}_S = \mathbf{U}^T \mathbf{\Lambda} \mathbf{U}$, where $\mathbf{\Lambda} \in \mathbb{R}^{N \times N}$ is a diagonal matrix with the eigenvalues of $\tilde{\mathbf{L}}_S$ on its nonzero entries, and

$\mathbf{U} \in \mathbb{R}^{N \times N}$ contains the corresponding eigenvectors on its columns. We perform K-Means clustering on the columns of $\tilde{\mathbf{U}} \in \mathbb{R}^{N \times K}$, *i.e.*, the matrix containing the first K eigenvectors on its columns. This procedure assigns each node i from $\mathcal{G}$ to one of the K clusters $c_i \in [1, \dots, K]$.

3.2. The HiERO architecture

Inspired by previous works in video understanding [18, 37], we encode an input video $\mathcal{V}$ as a *video graph* with N nodes $\mathcal{G} = (\mathbf{X}, \mathcal{E}, \mathbf{p})$, where $\mathbf{X} \in \mathbb{R}^{N \times D}$ is the node embeddings matrix, edge $e_{ij} \in \mathcal{E}$ connects nodes i and j if their temporal distance is smaller than a threshold τ and the attribute $\mathbf{p} \in \mathbb{R}^N$ encodes the temporal position of each node, *i.e.*, its timestamp in seconds. Each node represents a fixed-length segment of the video and the node embeddings are computed using a video features extractor, such as EgoVLP [28], from the segment frames. At training time, each video is also associated with a set of narrations, *i.e.*, concise textual descriptions of the actions represented in the video, denoted as $\mathcal{T}_\mathcal{V} = \{(n_i, t_i)\}_i$, where n_i and t_i are the textual narration and its corresponding timestamp. HiERO is built as an encoder-decoder architecture inspired by Graph U-Net [14]. The two branches share the same components but serve different roles. The *Temporal Encoder* $\mathcal{E}$ implements local temporal reasoning, hierarchically aggregating information between temporally close segments, while the *Function-Aware Decoder* $\mathcal{D}$ extends temporal reasoning to nodes that may be temporally distant but functionally similar, by connecting nodes belonging to the same thread. The architecture of HiERO is presented in Fig. 3.

Temporal Encoder. The *Temporal Encoder* $\mathcal{E}$ is implemented as a stack of N_l GNN-based blocks with temporal subsampling operations to map the input video graph $\mathcal{G}^{(0)}$ to a set of temporally coarsened representations:

$$\mathcal{E} : \mathcal{G}^{(0)} \rightarrow \{\mathcal{G}_e^{(1)}, \mathcal{G}_e^{(2)}, \dots, \mathcal{G}_e^{(N_l)}\}. \quad (1)$$

At the stage of the encoder at depth l, the temporal neighborhood of each node is defined as the set of all the nodes within a certain temporal distance d, adjusted for the depth i of the encoder stage. Each stage is composed of multiple TDGC [37] layers that implement temporal reasoning on the graph by combining the embedding of node i with a learnable projection of its neighbors $\mathcal{N}(i)$:

$$\mathbf{x}_j^{'} = \texttt{MLP}\left(\mathbf{x}_j^l\right) = \phi(\mathbf{W}_n^T \mathbf{x}_j^l + \mathbf{b}_n), \quad (2)$$

$$\mathbf{x}_i^{l+1} = \mathbf{W}_r^T \mathbf{x}_i^l + \underset{j \in \mathcal{N}(i)}{\texttt{mean}} \left(s_{ij} (\mathbf{w}_{ij} \odot \mathbf{x}_j^{'}) \right) + \mathbf{b}_r, \quad (3)$$

where $\mathbf{W}_n^T$ and $\mathbf{W}_r^T$ are learnable projection matrices, $\mathbf{b}_n$ and $\mathbf{b}_r$ are bias terms. s_{ij} and $\mathbf{w}_{ij}$ are used to rescale the contribution of each node depending on its temporal distance and are computed as:

$$s_{ij} = \texttt{sign}(\mathbf{p}_{[i]}^l - \mathbf{p}_{[j]}^l), \quad \mathbf{w}_{ij} = \texttt{MLP}(|\mathbf{p}_{[i]}^l - \mathbf{p}_{[j]}^l|). \quad (4)$$

Then, the nodes are subsampled to halve the temporal resolution of the graph and obtain $\mathcal{G}^{(l+1)}$, which is fed to the next layer of the encoder. Therefore, the encoder progressively extends the temporal context of the nodes, regardless of whether the actions performed are related or not.

Function-Aware Decoder. The *Function-Aware Decoder* $\mathcal{D}$ shares the same architecture of the encoder with one significant difference: instead of implementing message passing on the local temporal neighborhood of the nodes, each decoder stage first groups the graph nodes based on their functional similarity, *i.e.*, whether they represent functionally similar actions, and then implements temporal reasoning on each group separately. This procedure connects nodes that may be temporally distant but encode similar actions (*functional threads*), allowing the model to reason about long-term patterns not necessarily connected in time. The training process of HiERO (Sec. 3.2.1) explicitly pushes nodes that are assigned to the same cluster to be close in the features space and far from nodes assigned to other clusters.

First, each stage l of the decoder takes graph $\mathcal{G}_e^l$ from the corresponding temporal encoder stage via a lateral connection and the output of the upper layer of the decoder $\mathcal{G}_d^{l+1}$. The node features of $\mathcal{G}_d^{l+1}$ are then interpolated to match the temporal resolution of $\mathcal{G}_e^l$ and the two contributions are summed together. The resulting graph $\tilde{\mathcal{G}}_d^{l+1}$ is then fed to the *Cut & Match* module (Fig. 3), which partitions the graph into a set of K smaller graphs $\{\tilde{\mathcal{G}}_{d,1}^{l+1}, \dots, \tilde{\mathcal{G}}_{d,K}^{l+1}\}$, each corresponding to a group of functionally similar nodes, following the process described in Sec. 3.1. The tensor $\mathbf{c}^{l+1} \in [1, \dots, K]^n$ encodes the cluster assignment for each of the K sub-graphs obtained from $\tilde{\mathcal{G}}_d^{l+1}$. After this process, nodes that correspond to far apart segments of the video may be clustered together. We use TDGC to perform temporal reasoning into each partition separately and map the nodes back to the original graph to obtain $\mathcal{G}_d^{l+1}$, which becomes the input of the next decoder layer. As the number of nodes in the graph may grow rapidly with the size of the graph, we subsample the graph to a fixed and smaller size before processing with the *Cut & Match* module, compute the cluster assignments and propagate them back to the original nodes using a `1-NN` approach. More details on this approach in the appendix.

3.2.1. Training HiERO

We train HiERO to map video segments representing co-occurring actions close in the feature space (*video-narrations alignment loss* $\mathcal{L}_{vna}$) and to detect functional threads not necessarily close in time (*functional threads loss* $\mathcal{L}_{ft}$), for example in the case of interleaved activities, without using any specific supervision other than textual narrations. HiERO is trained with a combination of the two losses $\mathcal{L} = \mathcal{L}_{vna} + \mathcal{L}_{ft}$.

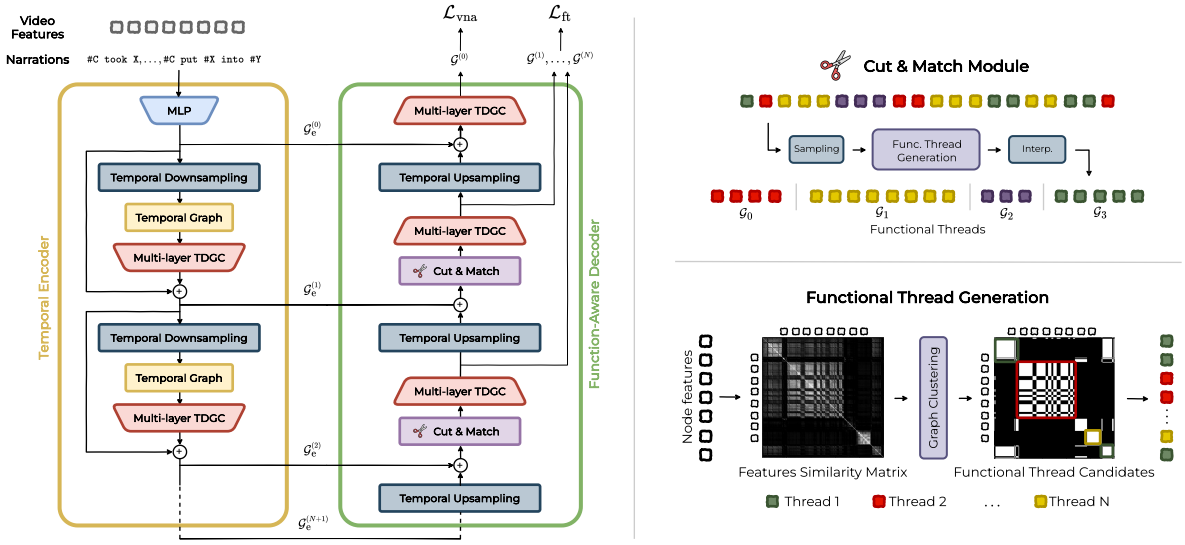


Figure 3. **Architecture of HiERO.** HiERO is designed as an encoder-decoder architecture to implement *Function-Aware video-text alignment*. The **Temporal Encoder** $\mathcal{E}$ performs temporal reasoning on graph representations of the input video at different scales, while the **Function-Aware Decoder** $\mathcal{D}$ recombines nodes in the video graph by matching segments that represent functional dependencies between the actions (*Cut & Match* module). HiERO is trained to align video segments with their corresponding textual narrations at the shallower layer, and to strengthen thread-aware clustering in deeper layers.

Video-narrations alignment. The *video-narrations alignment loss* $\mathcal{L}_{vna}$ encourages the network to map closer in the features space actions that typically occur together. $\mathcal{L}_{vna}$ is inspired by previous works in video-language pretraining [28, 38, 56] and is defined as a contrastive loss that pushes the node embeddings close to the text embeddings of the narrations that fall within a certain temporal window around the timestamp associated to the node (*positives*), while pushing apart narrations outside the window or appearing in other videos in the batch (*negatives*). Unlike previous works, like EgoVLP [28], which align each temporal segment with one single narration from the same window, ignoring the temporal context in which actions occur, our approach explicitly considers the co-occurrence of multiple actions in the temporal window covered by the node, making the embedding more context-aware. As a result, usually co-occurring actions have more similar embeddings that could be clustered more easily into high-level patterns.

Given a batch of B graphs $\{\mathcal{G}_1, \dots, \mathcal{G}_B\}$ with their corresponding narrations $\{\mathcal{T}_1, \dots, \mathcal{T}_B\}$, we define $\mathbf{V}_i \in \mathbb{R}^{N \times D_v}$ as the node embeddings of graph $\mathcal{G}_i$ at the output of the last decoder layer. The set of positive narrations for node j from graph $\mathcal{G}_i$ is defined as $\mathcal{P}_j = \{(n,t) \in \mathcal{T}_i \text{ s.t. } |p-t| \leq 2^\alpha\}$, where p is the timestamp associated to the node. Similarly, the negatives are defined as the narrations outside the window associated to the node or from other videos in the batch: $\mathcal{N}_j = \{(n,t) \in \mathcal{T}_i \text{ s.t. } 2^\alpha < |p-t| \leq 2^\beta\} \cap \mathcal{T}_{k,k\neq i}$. α and β control the size of the alignment window for positives and negatives sampling. Formally, the loss $\mathcal{L}_{vna}$ is defined as the sum of two symmetric contributions for *video-to-text* ($\mathcal{L}_{v2t}$) and *text-to-video* ($\mathcal{L}_{t2v}$) alignment:

$$\mathcal{L}_{v2t} = \frac{1}{B} \sum_{\mathbf{v}_j} \frac{\sum_{n \in \mathcal{P}_j} \exp(h_v(\mathbf{v}_j)^T h_t(\mathcal{F}(n))/\tau)}{\sum_{n \in \mathcal{P}_j \cup \mathcal{N}_j} \exp(h_v(\mathbf{v}_j)^T h_t(\mathcal{F}(n))/\tau)}, \quad (5)$$

where h_v and h_t are linear projections followed by L2-normalization to map the visual and textual embeddings in the same features space for alignment, $\mathcal{F}$ is a text features extractor, *e.g.*, BERT, and τ is a temperature parameter. The *text-to-video* loss ($\mathcal{L}_{t2v}$) is symmetrically defined in the same way.

Functional threads loss. Aligning the visual embeddings from larger temporal windows to their corresponding textual descriptions is more difficult. Using narrations is impractical as they are too fine-grained and the number of positive and negatives samples would grow rapidly with the depth of the network and the size of the alignment window. Other forms of *high-level* supervision, *e.g.*, video summaries, require huge annotation efforts. Instead, we apply video-narrations alignment only on the output of the decoder and introduce a contrastive regularization objective to make features at deeper layers belonging to the same functional thread more similar to each other. The *functional threads loss* $\mathcal{L}_{ft}$ leverages the graph partition assignments from the *Cut & Match* modules in the decoder and pushes closer to each other samples that are assigned to the same cluster, while pushing away samples from other clusters. Specifically, given the node embeddings $\mathbf{V}_i^l \in \mathbb{R}^{n \times D_v}$ at the output of the decoder for graph $\mathcal{G}_i$ with n nodes at depth

l, the $\mathcal{L}_{ft}$ is defined as:

$$\mathcal{L}_{ft} = \sum_{k=1}^{K} \sum_{\substack{i=1 \\ c_i=k}}^{n} \sum_{\substack{j=1 \\ c_i=c_j}}^{n} \frac{\exp(h_v(\mathbf{v}_i)^T h_v(\mathbf{v}_j)/\tau)}{\sum_{j'=1}^{n} \exp(h_v(\mathbf{v}_i)^T h_v(\mathbf{v}_{j'})/\tau)}, \quad (6)$$

where c_i represents the cluster assignment of node i.

3.3. Zero-shot procedural tasks

After training, HiERO can detect candidate procedure steps by clustering the output of the decoder at different granularities. This enables our approach to address different procedure learning tasks, including the segmentation of all the steps in the video (***procedure learning***), the temporal grounding of a step given its free-form textual description (***step grounding***) and the localization and classification of all the steps and sub-steps in a video (***step localization***), without any additional training.

Given the node embeddings $\mathbf{V}_i^l \in \mathbb{R}^{n \times D_v}$ of graph $\mathcal{G}_i$ with n nodes at depth l, we apply the clustering method proposed in Sec. 3 to assign each node in the graph to one out of K possible clusters: $\mathbf{c}^l \in [1, \dots, K]^n$. The cluster assignments are then upsampled to match the frame rate of the input video $\mathbf{c} = \texttt{UP}(\mathbf{c}^l)$. For *procedure learning*, $\mathbf{c}$ represents the step assignments for each segment. For other tasks, we map the output features of the decoder using h_v, apply the clustering algorithm and aggregate features from consecutive segments that are assigned to the same step to obtain a set of M candidate step embeddings $\{\mathbf{F}_1, \dots, \mathbf{F}_M\}$ with $\mathbf{F}_i \in \mathbb{R}^{D_v}$. Short candidate segments are discarded as background. For *step localization*, given a textual taxonomy consisting of S step labels and the corresponding textual embeddings $\mathbf{T} \in \mathbb{R}^{S \times D_t}$, we assign each step candidate the label y_i that maximizes the cosine similarity between the average visual features of the segment $\mathbf{f}_i$ and the step label embedding: $y_i = \arg\max_j \mathbf{f}_i^T \mathbf{t}_j / \|\mathbf{f}_i\| \|\mathbf{t}_j\|$. For *step grounding*, we extract an embedding $\mathbf{t}$ from the textual query and select the candidate steps based on cosine similarity between their visual features and the query embedding. More details in the appendix.

4. Experiments

We train HiERO on EgoClip [28], a curated set of 3.8M clip-text pairs obtained from Ego4D textual narrations, using pre-extracted features from several backbones, *i.e.*, Omnivore [15], EgoVLP [28] and LaViLa [56], showing that HiERO can be easily applied to different backbones. For EgoVLP and LaViLa we reuse their text encoders when training HiERO, while for Omnivore we start from a pre-trained DistillBERT [42] model and fine-tune it during training. We train HiERO for 15 epochs, using batch size 8 and learning rate 1×10^{-5} with linear warmup for the first 5 epochs and a cosine annealing schedule. Training takes less than 20 GPU hours. More details in the appendix.

Method	EgoMCQ		EgoNLQ			
	Accuracy (%)		mIOU@0.3		mIOU@0.5	
	Inter	Intra	R@1	R@5	R@1	R@5
Omnivore [15]† (CVPR'22)	–	–	6.56	12.55	3.59	7.90
SlowFast [13] (ICCV'19)	–	–	5.45	10.74	3.12	6.63
EgoVLP [28] (NIPS'22)	90.6	57.2	10.84	18.84	6.81	13.45
HierVL [2] (CVPR'23)	90.5	52.4	–	–	–	–
LaViLa [56] (CVPR'23)	94.5	63.1	12.05	22.38	7.43	15.44
EgoVLPv2 [38] (ICCV'23)	91.0	60.9	12.95	**23.80**	7.91	**16.11**
Ours (Omnivore)	90.1	53.4	10.27	18.20	6.01	12.52
Ours (EgoVLP)	91.6	59.6	11.41	19.67	7.05	13.91
Ours (LaViLa)	**94.6**	**64.4**	**13.35**	21.12	**8.08**	15.31

Table 1. **Results on EgoMCQ and EgoNLQ's validation set**, using VSLNet [55] as grounding head for the latter. †Reproduced.

Evaluation benchmarks. We evaluate our approach on several egocentric vision benchmarks to validate its effectiveness in different scenarios. Specifically, we validate the video-text alignment components of HiERO on **EgoMCQ** [28], a set of 39K *text-to-video* multiple-choice questions derived from Ego4D narrations, and **EgoNLQ**, a natural language queries benchmark that aims to localize the segment of a video (start and end timestamps) answering a given textual query. For Procedure Learning, we evaluate HiERO on **EgoProceL** [4], a large scale benchmark with 62 hours of procedural videos from a set of 16 different tasks, and on the Step Grounding and Step Localization tasks from **Goal-Step** [47], a subset of Ego4D featuring procedure annotations from a taxonomy of 514 fine-grained steps and substeps. The design of HiERO allows to address most of these tasks in a completely *zero-shot* setting.

4.1. Quantitative Results

4.1.1. Video-Text Alignment on EgoMCQ and EgoNLQ

We evaluate HiERO on EgoMCQ [28] and EgoNLQ [17] to validate its video-text alignment capabilities and to show that reasoning on functional threads at different scales can support various tasks (Table 1). EgoMCQ is a multiple-choice *text-to-video* retrieval task where the goal is to select the video clip that matches a given textual description among five possible candidates. Results are measured in terms of *inter* (options are from different videos) and *intra* accuracy (options are from the same video). EgoNLQ aims at localizing the temporal segment of a video that answers a textual query, *e.g.*, *Where did I put X?* or *Where is object X before / after event Y?*. These queries require strong temporal and causal understanding of the interactions between different objects and actions in the video. Performance is measured with Recall at different IoU thresholds between the predicted and the ground truth segments. For this task, we follow previous approaches [17, 28, 38, 56] and train a VSLNet [55] grounding head on top of the features at the output of the decoder of HiERO.

Our window-based alignment loss encourages HiERO to learn functional dependencies between actions, while clustering groups together similar actions at different scales and

Method	Average		CMU-MMAC [10]		EGTEA [27]		MECCANO [40]		EPIC-Tents [20]		PC Ass. [4]		PC Disass. [4]	
	F1	IoU	F1	IoU	F1	IoU	F1	IoU	F1	IoU	F1	IoU	F1	IoU
Random [8] (NeurIPS'24)	14.8	6.1	15.7	5.9	15.3	4.6	13.4	5.3	14.1	6.5	15.1	7.2	15.3	7.1
CnC [4] (ECCV'22)	22.0	10.7	22.7	11.1	21.7	9.5	18.1	7.8	17.2	8.3	25.1	12.8	27.0	14.8
GPL-2D [5] (WACV'24)	22.0	11.9	21.8	11.7	23.6	14.3	18.0	8.4	17.4	8.5	24.0	12.6	27.4	15.9
GPL [5] (WACV'24)	25.6	13.9	31.7	17.9	27.1	16.0	20.7	10.0	19.8	9.1	27.5	15.2	26.7	15.2
OPEL [8] (NeurIPS'24)	32.0	16.3	36.5	18.8	29.5	13.2	39.2	20.2	20.7	10.6	33.7	17.9	32.2	16.9
Omnivore	39.1	22.0	44.7	26.8	37.1	19.2	36.0	19.0	40.8	21.9	35.7	21.5	40.3	23.5
Ours (Omnivore)	44.0	24.5	47.2	27.7	**39.7**	**19.9**	**41.6**	**22.1**	**45.3**	**24.3**	43.7	25.1	46.3	27.9
EgoVLP	40.0	21.9	49.2	31.0	36.6	18.3	33.1	16.1	37.4	19.2	38.2	20.8	45.4	25.6
Ours (EgoVLP)	**44.5**	**25.3**	**53.5**	**34.0**	**39.7**	19.6	39.8	20.3	39.0	20.3	**44.9**	**25.6**	**49.9**	**32.1**

Table 2. **Comparison with the state-of-the-art on the EgoProceL benchmark [4]**. Performance is evaluated in terms of F1 score and IoU w.r.t. ground truth key-steps, using a fixed number of predicted key-steps ($k = 7$) for a fair comparison to the previous approaches.

over a long temporal horizon. Together, these objectives are effective to discriminate between similar short-term actions, which is critical for EgoMCQ, as well as to capture long-range causal and temporal dependencies in the video, which is essential for EgoNLQ. Unlike other backbones that extract features from a short temporal window and rely entirely on the grounding head for high-level reasoning, our features inherently capture a broader semantic understanding of the video. In both benchmarks, HiERO significantly improves the SOTA, regardless of the backbone (+1.3% on intra accuracy on EgoMCQ and Top-1 Recall at IoU = 0.3 on EgoNLQ when using LaViLa features). Remarkably, HiERO achieves good results even with Omnivore features, despite not being trained end-to-end on Ego4D.

4.1.2. Procedure Learning on EgoProceL

We evaluate HiERO on EgoProceL [4] in *zero-shot*, using visual features extracted from the Omnivore and EgoVLP backbones. Following the original evaluation protocol [4], we compute frame-wise step assignments and match them with the ground truth using the Hungarian algorithm [24]. Performance is measured in terms of F1 score and IoU with respect to the ground truth key-steps. More details in the appendix. Compared to previous works in this setting that are based on matching visual segments between pairs of videos representing the same task, *e.g.*, CnC [4], GPL [4] and OPEL [8], our approach is fundamentally different and does not require any additional supervision. Indeed, we evaluate on this benchmark the ability of HiERO to group together parts of the video that correspond to the same high-level activity by leveraging their functional similarity, even though it was not trained explicitly to identify procedure steps inside a video. We compare HiERO with the SOTA in Table 2, using a fixed number of key-steps to predict ($k = 7$) for a fair comparison with previous approaches that share the same assumption. Using our clustering approach in combination with Omnivore and EgoVLP features is already particularly effective in detecting the procedure steps (+7.1% and +8.0% respectively compared to the previous state-of-the-art OPEL [8]), supporting the intuition that steps can emerge as clusters of similar actions [43]. HiERO significantly improves over these baselines (+4.9% and +4.5% respectively), showing that i) procedure steps can emerge by actions clustering without the need of specific supervision, ii) HiERO can generalize to novel procedural tasks that were not present in Ego4D. We present an in depth comparison of our baselines and OPEL in the appendix.

Method	Approach	mIoU@0.3		mIoU@0.5	
		R@1	R@5	R@1	R@5
Omnivore [47]	Supervised	12.02	19.99	7.71	14.17
Ours (Omnivore)	Supervised	13.02	21.81	8.59	15.98
EgoVLP	Supervised	15.43	25.91	10.95	19.77
Ours (EgoVLP)	Supervised	**15.64**	**26.01**	**11.14**	**20.08**
EgoVLP	Zero-Shot	10.73	24.70	7.38	16.53
Ours (Omnivore)	Zero-Shot	9.29	22.89	6.24	15.05
Ours (EgoVLP)	Zero-Shot	**11.57**	**27.41**	**7.87**	**18.70**

Table 3. **Step-Grounding on Ego4D Goal-Step [47]**. In the *Supervised* setting, we compare different features extractors, including HiERO, using VSLNet [55] as grounding head. In the *Zero-Shot* setting, we adopt the clustering approach of HiERO.

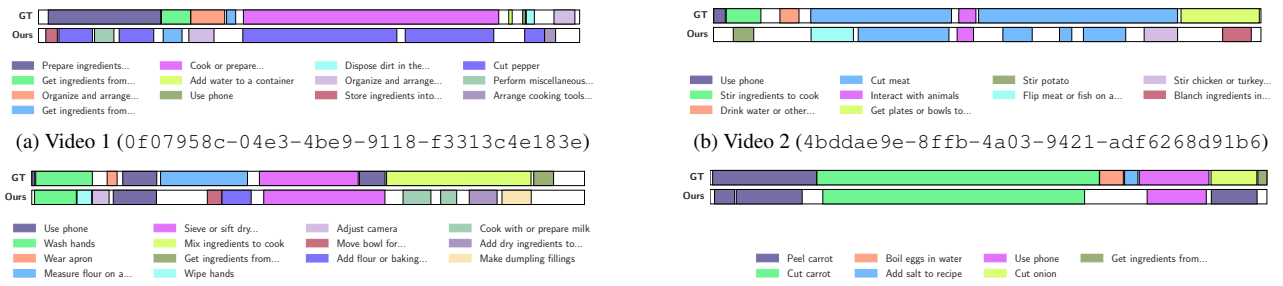
Method	Approach	mAP @ IoU					
		0.1	0.2	0.3	0.4	0.5	Avg
Omnivore [47]	Supervised	–	–	–	–	–	10.3
EgoOnly [47]	Supervised	–	–	–	–	–	**13.6**
EgoVLP	Supervised	13.3	12.3	11.2	10.1	8.7	11.1
Ours (EgoVLP)	Supervised	14.2	13.2	12.2	10.9	9.6	12.0
EgoVLP	Zero-Shot	11.8	9.7	8.3	6.7	5.1	8.3
Ours (EgoVLP)	Zero-Shot	12.0	10.0	8.8	7.3	5.6	8.7

Table 4. **Step Localization on Ego4D Goal-Step [47]**, in *supervised* and *zero-shot* settings. In the *supervised* setting we use ActionFormer [54] as localization head, while HiERO detects steps directly on the output features of the decoder using zero-shot matching with the steps taxonomy.

4.1.3. Step Grounding and Localization on Goal-Step

We evaluate HiERO on the Step Grounding and Step Localization tasks from Ego4D Goal-Step [47], demonstrating its ability to localize and classify procedural steps.

Step Grounding. This task aims to localize a procedure step given its description in natural language. Performance is measured with Recall at different IoU thresholds, as for EgoNLQ. The supervised baseline proposed in [47] leverages VSLNet [55] as grounding head on top of the Om-



(a) Video 1 (`0f07958c-04e3-4be9-9118-f3313c4e183e`)

(b) Video 2 (`4bddae9e-8ffb-4a03-9421-adf6268d91b6`)

(c) Video 3 (`603a427f-9191-4ca4-a1b0-dd3c5e7fda70`)

(d) Video 4 (`64b355f3-ef49-4990-8622-9e9eef68b495`)

Figure 4. **Zero-Shot Localization results on Ego4D Goal-Step, showing some of the HiERO's success and failure cases.** We observe that many failure cases of HiERO are related to the ambiguous granularity of the step annotations in the dataset. In Fig. 4a, HiERO confuses the step *Cook or prepare the vegetables* with the closely related *Cut the pepper*. In Fig. 4c, HiERO correctly identifies many steps but confuses *Mix ingredients to cook* with some of its possible sub-steps, *e.g.*, *Cook with or prepare milk*.

nivore pre-extracted features. Instead, we adapt HiERO to this task by clustering the video segments and selecting as prediction candidates the segment whose average visual features are most similar to the textual features of the query step. This allows to address the grounding task in *zero-shot* without any additional training. We also evaluate the performance of HiERO when used as a feature extractor in combination with VSLNet. Table 3 shows that HiERO consistently outperforms the Omnivore and EgoVLP baselines in the supervised setting. In *zero-shot*, HiERO beats the supervised counterpart on Top-5 Recall and achieves results close to the SOTA on the other metrics.

Step Localization. This task aims to predict triplets `(start time, end time, label)` for all the procedure steps and substeps in the video. We adapt HiERO to this task by clustering the output features to localize the steps and use the similarity between the visual and the textual features of the steps taxonomy to predict their labels (Table 4). We compare HiERO with the two official baselines from Goal-Step [47], which train an ActionFormer [54] localization head on top of Omnivore [15] and EgoOnly [51] features. Performance is evaluated in terms of mAP at different IoU thresholds. Notably, unlike other approaches that generate per-segment predictions and apply Soft-NMS [6] to filter overlaps, HiERO produces non-overlapping step candidates and does not require any post-processing. Remarkably, the zero-shot results of HiERO demonstrate that our clustering approach effectively identifies action clusters, which are well aligned with the steps taxonomy. Compared to supervised approaches that learn a direct mapping between the video and the procedure steps, we argue that the steps detected by HiERO emerge as composition of low-level patterns that are clustered together.

4.2. Ablation on the HiERO components

We analyze in Table 5 the impact of the different components of HiERO, using the EgoVLP backbone, on three significant tasks, namely EgoMCQ [28], Procedure Learning on EgoProceL [4] and Step Grounding on Goal-Step [47].

Align Loss	Func. Th. Cluster	Func Th. Loss	EgoMCQ Inter	EgoMCQ Intra	EgoProceL F1	EgoProceL IoU	Step-Grounding R@1	Step-Grounding R@5
✗	✗	✗	90.6	57.2	40.0	21.9	10.73	24.70
✓	✗	✗	**91.8**	59.5	43.8	24.1	11.27	27.35
✓	✓	✗	**91.8**	**59.6**	43.3	24.2	11.44	27.12
✓	✓	✓	91.6	**59.6**	**44.5**	**25.3**	**11.57**	**27.41**

Table 5. **Ablation of the different components of HiERO** on EgoMCQ, EgoProceL and Goal-Step, using EgoVLP features.

Compared to the baseline, the alignment loss $\mathcal{L}_{vna}$ significantly improves performance on all the tasks, demonstrating that the context-aware features of HiERO effectively support various understanding tasks, particularly procedural ones. Training-time threads clustering has a more mild impact. However, the introduction of the *functional threads* loss $\mathcal{L}_{ft}$ effectively guides the clustering process by encouraging samples within the same cluster to be closer in feature space, leading to better performance.

4.3. Qualitative results

We show in Fig. 4 some success and failure cases of HiERO in the *zero-shot* Step Localization task on Goal-Step. We observe that many failure cases of our approach are related to the ambiguous granularity of the step labels in the ground truth, which leads to confusion between steps that could be either steps or sub-steps, *e.g.*, *Cook or prepare the vegetables* and *Cut the pepper* in Fig. 4a. We provide additional qualitative results in the appendix.

5. Conclusions

In this paper, we discuss the relevance of learning about the hierarchical structure of human behavior collected in egocentric videos. We propose HiERO, a weakly-supervised method able to fully exploit functional threads to enhance reasoning capabilities for multiple downstream tasks. HiERO delivers state of the art performance in zero-shot for procedural learning tasks, proving the effectiveness and importance of using functional reasoning at multiple levels. HiERO features proved their suitability for video-text alignment tasks, outperforming foundational models features.

Acknowledgments

This study was carried out within the FAIR - Future Artificial Intelligence Research and received funding from the European Union Next-GenerationEU (PIANO NAZIONALE DI RIPRESA E RESILIENZA (PNRR) – MISSIONE 4 COMPONENTE 2, INVESTIMENTO 1.3 – D.D. 1555 11/10/2022, PE00000013). This manuscript reflects only the authors' views and opinions, neither the European Union nor the European Commission can be considered responsible for them. We acknowledge the CINECA award under the ISCRA initiative, for the availability of high performance computing resources and support.

References

[1] Triantafyllos Afouras, Effrosyni Mavroudi, Tushar Nagarajan, Huiyu Wang, and Lorenzo Torresani. Ht-step: Aligning instructional articles with how-to videos. In *NeurIPS*, 2023. 2

[2] Kumar Ashutosh, Rohit Girdhar, Lorenzo Torresani, and Kristen Grauman. Hiervl: Learning hierarchical video-language embeddings. In *CVPR*, 2023. 1, 2, 6

[3] Kumar Ashutosh, Santhosh Kumar Ramakrishnan, Triantafyllos Afouras, and Kristen Grauman. Video-mined task graphs for keystep recognition in instructional videos. *NeurIPS*, 2023. 2

[4] Siddhant Bansal, Chetan Arora, and CV Jawahar. My view is the best view: Procedure learning from egocentric videos. In *ECCV*, 2022. 1, 2, 6, 7, 8

[5] Siddhant Bansal, Chetan Arora, and CV Jawahar. United we stand, divided we fall: Unitygraph for unsupervised procedure learning from videos. In *WACV*, 2024. 2, 7

[6] Navaneeth Bodla, Bharat Singh, Rama Chellappa, and Larry S Davis. Soft-nms–improving object detection with one line of code. In *ICCV*, 2017. 8

[7] Matthew M Botvinick, Yael Niv, and Andew G Barto. Hierarchically organized behavior and its neural foundations: A reinforcement learning perspective. *Cognition*, 2009. 1

[8] Sayeed Shafayet Chowdhury, Soumyadeep Chandra, and Kaushik Roy. Opel: Optimal transport guided procedure learning. In *NeurIPS*, 2024. 1, 2, 7

[9] Richard P Cooper and Tim Shallice. Hierarchical schemas and goals in the control of sequential behavior. *Psychological Review*, 2006. 1

[10] Fernando De la Torre, Jessica Hodgins, J Montano, S Valcarcel, R Forcada, and J Macey. Carnegie mellon university multimodal activity (cmu-mmac) database, 2008. 7

[11] Nikita Dvornik, Isma Hadji, Ran Zhang, Konstantinos G Derpanis, Richard P Wildes, and Allan D Jepson. Stepformer: Self-supervised step discovery and localization in instructional videos. In *CVPR*, 2023. 2

[12] Ehsan Elhamifar and Dat Huynh. Self-supervised multi-task procedure learning from instructional videos. In *ECCV*, 2020. 2

[13] Christoph Feichtenhofer, Haoqi Fan, Jitendra Malik, and Kaiming He. Slowfast networks for video recognition. In *ICCV*, 2019. 6

[14] Hongyang Gao and Shuiwang Ji. Graph u-nets. In *ICML*, 2019. 2, 4

[15] Rohit Girdhar, Mannat Singh, Nikhila Ravi, Laurens van der Maaten, Armand Joulin, and Ishan Misra. Omnivore: A single model for many visual modalities. In *CVPR*, 2022. 1, 6, 8

[16] Gabriele Goletto, Tushar Nagarajan, Giuseppe Averta, and Dima Damen. Amego: Active memory from long egocentric videos. In *ECCV*, 2024. 2

[17] Kristen Grauman, Andrew Westbury, Eugene Byrne, Zachary Chavis, Antonino Furnari, Rohit Girdhar, Jackson Hamburger, Hao Jiang, Miao Liu, Xingyu Liu, et al. Ego4d: Around the world in 3,000 hours of egocentric video. In *CVPR*, 2022. 3, 6

[18] Yifei Huang, Yusuke Sugano, and Yoichi Sato. Improving action segmentation via graph-based temporal reasoning. In *CVPR*, 2020. 4

[19] Md Mohaiminul Islam, Ngan Ho, Xitong Yang, Tushar Nagarajan, Lorenzo Torresani, and Gedas Bertasius. Video recap: Recursive captioning of hour-long videos. In *CVPR*, 2024. 2

[20] Youngkyoon Jang, Brian Sullivan, Casimir Ludwig, Iain Gilchrist, Dima Damen, and Walterio Mayol-Cuevas. Epic-tent: An egocentric video dataset for camping tent assembly. In *ICCVW*, 2019. 7

[21] Baoxiong Jia, Ting Lei, Song-Chun Zhu, and Siyuan Huang. Egotaskqa: Understanding human tasks in egocentric videos. In *NeurIPS*, 2022. 2

[22] Matthew Kowal, Mennatullah Siam, Md Amirul Islam, Neil DB Bruce, Richard P Wildes, and Konstantinos G Derpanis. Quantifying and learning static vs. dynamic information in deep spatiotemporal networks. *IEEE TPAMI*, 2024. 2

[23] H. Kuehne, A. B. Arslan, and T. Serre. The language of actions: Recovering the syntax and semantics of goal-directed human activities. In *CVPR*, 2014. 3

[24] Harold W Kuhn. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 1955. 7

[25] Sateesh Kumar, Sanjay Haresh, Awais Ahmed, Andrey Konin, M Zeeshan Zia, and Quoc-Huy Tran. Unsupervised action segmentation by joint representation learning and online clustering. In *CVPR*, 2022. 2

[26] Jun Li and Sinisa Todorovic. Set-constrained viterbi for set-supervised action segmentation. In *CVPR*, 2020. 2

[27] Yin Li, Miao Liu, and James M Rehg. In the eye of beholder: Joint learning of gaze and actions in first person video. In *ECCV*, 2018. 7

[28] Kevin Qinghong Lin, Jinpeng Wang, Mattia Soldan, Michael Wray, Rui Yan, Eric Z XU, Difei Gao, Rong-Cheng Tu, Wenzhe Zhao, Weijie Kong, et al. Egocentric video-language pretraining. In *NeurIPS*, 2022. 2, 4, 5, 6, 8

[29] Xudong Lin, Fabio Petroni, Gedas Bertasius, Marcus Rohrbach, Shih-Fu Chang, and Lorenzo Torresani. Learning to recognize procedural activities with distant supervision. In *CVPR*, 2022. 2

[30] Karttikeya Mangalam, Raiymbek Akshulakov, and Jitendra Malik. Egoschema: A diagnostic benchmark for very long-form video language understanding. In *NeurIPS*, 2023. 2

[31] Effrosyni Mavroudi, Triantafyllos Afouras, and Lorenzo Torresani. Learning to ground instructional articles in videos through narrations. In *ICCV*, 2023. 2

[32] Luke Melas-Kyriazi, Christian Rupprecht, Iro Laina, and Andrea Vedaldi. Deep spectral methods: A surprisingly strong baseline for unsupervised semantic segmentation and localization. In *CVPR*, 2022. 2

[33] Tushar Nagarajan, Christoph Feichtenhofer, and Kristen Grauman. Grounded human-object interaction hotspots from video. In *ICCV*, 2019. 2

[34] Tushar Nagarajan, Yanghao Li, Christoph Feichtenhofer, and Kristen Grauman. Ego-topo: Environment affordances from egocentric video. In *CVPR*, 2020. 2

[35] Zwe Naing and Ehsan Elhamifar. Procedure completion by learning from partial summaries. In *BMVC*, 2020. 2

[36] Simone Alberto Peirone, Francesca Pistilli, Antonio Alliegro, and Giuseppe Averta. A backpack full of skills: Egocentric video understanding with diverse task perspectives. In *CVPR*, 2024. 2

[37] Simone Alberto Peirone, Francesca Pistilli, Antonio Alliegro, Tatiana Tommasi, and Giuseppe Averta. Hier-egopack: Hierarchical egocentric video understanding with diverse task perspectives. *arXiv preprint arXiv:2502.02487*, 2025. 4

[38] Shraman Pramanick, Yale Song, Sayan Nag, Kevin Qinghong Lin, Hardik Shah, Mike Zheng Shou, Rama Chellappa, and Pengchuan Zhang. Egovlpv2: Egocentric video-language pre-training with fusion in the backbone. In *ICCV*, 2023. 1, 2, 5, 6

[39] Will Price, Carl Vondrick, and Dima Damen. Unweavenet: Unweaving activity stories. In *CVPR*, 2022. 1, 2

[40] Francesco Ragusa, Antonino Furnari, and Giovanni Maria Farinella. Meccano: A multimodal egocentric dataset for humans behavior understanding in the industrial-like domain. *CVIU*, 2023. 7

[41] Alexander Richard, Hilde Kuehne, Ahsan Iqbal, and Juergen Gall. Neuralnetwork-viterbi: A framework for weakly supervised video learning. In *CVPR*, 2018. 2

[42] V Sanh. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019. 6

[43] Saquib Sarfraz, Naila Murray, Vivek Sharma, Ali Diba, Luc Van Gool, and Rainer Stiefelhagen. Temporally-weighted hierarchical clustering for unsupervised action segmentation. In *CVPR*, 2021. 1, 2, 7

[44] Luigi Seminara, Giovanni Maria Farinella, and Antonino Furnari. Differentiable task graph learning: Procedural activity representation and online mistake detection from egocentric videos. In *NeurIPS*, 2024. 2, 3

[45] Yuhan Shen and Ehsan Elhamifar. Progress-aware online action segmentation for egocentric procedural task videos. In *CVPR*, 2024. 2

[46] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE TPAMI*, 2000. 2

[47] Yale Song, Eugene Byrne, Tushar Nagarajan, Huiyu Wang, Miguel Martin, and Lorenzo Torresani. Ego4d goal-step: Toward hierarchical understanding of procedural activities. In *NeurIPS*, 2024. 1, 2, 6, 7, 8

[48] Sebastian Stein and Stephen J McKenna. Combining embedded accelerometers with computer vision for recognizing food preparation activities. In *Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing*, 2013. 3

[49] Zhan Tong, Yibing Song, Jue Wang, and Limin Wang. Videomae: Masked autoencoders are data-efficient learners for self-supervised video pre-training. In *NeurIPS*, 2022. 1

[50] Ulrike Von Luxburg. A tutorial on spectral clustering. *Statistics and computing*, 17:395–416, 2007. 3

[51] Huiyu Wang, Mitesh Kumar Singh, and Lorenzo Torresani. Ego-only: Egocentric action detection without exocentric transferring. In *ICCV*, 2023. 8

[52] Xudong Wang, Ishan Misra, Ziyun Zeng, Rohit Girdhar, and Trevor Darrell. Videocutler: Surprisingly simple unsupervised video instance segmentation. In *CVPR*, 2024. 2

[53] Yangtao Wang, Xi Shen, Yuan Yuan, Yuming Du, Maomao Li, Shell Xu Hu, James L Crowley, and Dominique Vaufreydaz. Tokencut: Segmenting objects in images and videos with self-supervised transformer and normalized cut. *IEEE TPAMI*, 2023. 2

[54] Chen-Lin Zhang, Jianxin Wu, and Yin Li. Actionformer: Localizing moments of actions with transformers. In *ECCV*, 2022. 7, 8

[55] Hao Zhang, Aixin Sun, Wei Jing, and Joey Tianyi Zhou. Span-based localizing network for natural language video localization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020. 6, 7

[56] Yue Zhao, Ishan Misra, Philipp Krähenbühl, and Rohit Girdhar. Learning video representations from large language models. In *CVPR*, 2023. 1, 2, 5, 6

[57] Yiwu Zhong, Licheng Yu, Yang Bai, Shangwen Li, Xueting Yan, and Yin Li. Learning procedure-aware video representation from instructional videos and their narrations. In *CVPR*, 2023. 2

[58] Honglu Zhou, Roberto Martín-Martín, Mubbasir Kapadia, Silvio Savarese, and Juan Carlos Niebles. Procedure-aware pretraining for instructional video understanding. In *CVPR*, 2023. 2

[59] Luowei Zhou, Chenliang Xu, and Jason Corso. Towards automatic learning of procedures from web instructional videos. In *AAAI*, 2018. 2

[60] Dimitri Zhukov, Jean-Baptiste Alayrac, Ramazan Gokberk Cinbis, David Fouhey, Ivan Laptev, and Josef Sivic. Cross-task weakly supervised learning from instructional videos. In *CVPR*, 2019. 2