Hier-EgoPack: Hierarchical Egocentric Video Understanding with Diverse Task Perspectives

Hier-EgoPack: Hierarchical Egocentric Video Understanding With Diverse Task Perspectives

Simone Alberto Peirone, Francesca Pistilli, Antonio Alliegro, Tatiana Tommasi, and Giuseppe Averta

***Abstract*—Our comprehension of video streams depicting human activities is naturally multifaceted: in just a few moments, we can grasp what is happening, identify the relevance and interactions of objects in the scene, and forecast what will happen soon, everything all at once. To endow autonomous systems with such a holistic perception, learning how to correlate concepts, abstract knowledge across diverse tasks, and leverage tasks synergies when learning novel skills is essential. A significant step in this direction is EgoPack, a unified framework for understanding human activities across diverse tasks with minimal overhead. EgoPack promotes information sharing and collaboration among downstream tasks, essential for efficiently learning new skills. In this paper, we introduce Hier-EgoPack, which advances EgoPack by enabling reasoning also across diverse temporal granularities, which expands its applicability to a broader range of downstream tasks. To achieve this, we propose a novel hierarchical architecture for temporal reasoning equipped with a GNN layer specifically designed to tackle the challenges of multi-granularity reasoning effectively. We evaluate our approach on multiple Ego4D benchmarks involving both clip-level and frame-level reasoning, demonstrating how our hierarchical unified architecture effectively solves these diverse tasks simultaneously.**

***Index Terms*—Egocentric vision, video understanding, multi-task learning.**

I. Introduction

OUR daily activities are extremely complex and diverse, yet humans have the extraordinary ability to perceive, reason, and plan their actions almost entirely from visual inputs. For instance, when observing someone at a kitchen counter with a pack of flour and a jug of water, we can infer they are kneading dough (*reasoning about current activity*). We might predict that their next step will involve mixing flour with water (*reasoning about the future*) to obtain the dough (*reasoning about implications*), maybe with the ultimate goal of preparing some bread (*reasoning about long-range activities*). Mastering such "skills" requires analyzing varying portions of the video and reasoning at different levels of temporal granularity. Long-term activities require analysis of a broader context over extended clips, while finer details, such as distinguishing when someone shifts from measuring flour to pouring water, call for reasoning at a frame level. Such holistic reasoning, which is natural for humans, poses a significant challenge for artificial intelligence systems. The difficulty lies in integrating various levels of reasoning, from low-level actions to high-level activity understanding, into a unified framework, while uncovering and leveraging the underlying semantic relationships between these skills to efficiently learn new ones by building on prior knowledge.

Current research trends in human activity understanding predominantly focus on creating several, hyper-specialized, models. This approach splits the understanding of human activities into distinct skills (i.e., tasks), for which each model is independently trained to rely only on "task-specific" clues for prediction [1], [2], [3]. However, this approach overlooks that different tasks may share similar or complementary reasoning patterns, i.e., looking at the same video portion from different *perspectives*.

To leverage the interplay between such different task perspectives, a first strategy might involve Multi-Task Learning (MTL), exploiting the intuition that knowledge sharing between tasks may be beneficial for each of them. However, MTL suffers of some limitations [4], mainly related to negative interferences between tasks, making it difficult to exploit task synergies effectively. In addition, all task annotations must be available at training time, which hinders the extension of MTL models to novel tasks at a later point in time.

In the context of human behavior understanding, usually inferred from videos collected in first person view, different tasks typically require closely related reasoning, resulting in a strong correlation between them. Consequently, studying and leveraging these inter-task synergies becomes particularly interesting.

In this scenario, the EgoT2 framework [5] first explored how various egocentric video tasks can mutually benefit each other. EgoT2 builds a collection of different models, one for each task, and learns to translate task-specific cues across tasks. However, although this approach fosters positive interactions between tasks, it has significant limitations: i) the primary task should be "known" at training time and present within the task-specific model collection, ii) it necessitates an extensive pretraining process and iii) it is inefficient as it relies on task-specific models instead of building transferable knowledge abstractions.

We argue that an important key to advancing the learning capabilities of intelligent systems and moving closer to more human-like reasoning lies not only in sharing information across

Received 20 December 2024; revised 30 September 2025; accepted 5 October 2025. Date of publication 14 October 2025; date of current version 9 January 2026. The work of Antonio Alliegro and Tatiana Tommasi was supported by the EU project ELSA - European Lighthouse on Secure and Safe AI under Grant 101070617. This work was supported by 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, through FAIR – Future Artificial Intelligence Research. Recommended for acceptance by G. Kim. *(Corresponding author: Simone Alberto Peirone.)*

The authors are with the Department of Control and Computer Engineering, Politecnico di Torino, 10129 Turin, Italy (e-mail: simone.peirone@polito.it; francesca.pistilli@polito.it; antonio.alliegro@polito.it; tatiana.tommasi@polito.it; giuseppe.averta@polito.it).

Project Page: sapeirone.github.io/hier-egopack.

Digital Object Identifier 10.1109/TPAMI.2025.3621326

tasks, but also in abstracting task-specific knowledge to make it reusable for learning novel tasks. To enable this, we recently proposed EgoPack [6], a first effort in knowledge abstraction and sharing for egocentric videos understanding. This method is able to exploit a set of known tasks (*support tasks*), each one able to interpret an input stream according to its own task-specific perspective, to learn reusable knowledge abstractions that can aid in the learning of a *novel task*. Such task perspectives are encoded in the form of prototypes, collected in a single step from the pretraining of a multi-task network. However, EgoPack implements limited temporal reasoning and, due to its flat architecture, cannot perform reasoning at different levels of granularity. Notably, egocentric videos cover a wide range of tasks spanning diverse temporal scales, from sub-second actions to extended, long-range activities. While some tasks, such as action recognition and long-term anticipation, focus on fixed short segments, others, like temporal action localization, demand a more adaptive approach to deal with longer activities. As the temporal span of these tasks increases, developing a robust understanding of the sequential order of events, a concept known as *sense of time* [7], becomes essential.

To address these challenges, we introduce Hier-EgoPack, an extension of EgoPack [6], specifically designed to maximize positive interaction across tasks with different temporal granularity, while still using a unified architecture and minimizing task-specific weights and tuning. To achieve this, we present a hierarchical architecture that progressively learns more comprehensive representations of the input video, capturing both fine-grained details and broad contextual patterns. A key aspect of this hierarchical design is effectively reasoning on temporal dependencies and consequentiality of actions, encompassing both past and future contexts. To address this, we develop a novel GCN layer, hereinafter called Temporal Distance Gated Convolution (TDGC), specifically designed to encode these temporal relationships effectively.

We demonstrate the effectiveness and efficiency of our approach on Ego4D [8], a large-scale egocentric vision dataset. To summarize, our main contributions are:

1) We introduce a unified video understanding architecture to learn multiple egocentric vision tasks with different temporal granularity, while requiring minimal task-specific overhead;
2) We present Temporal Distance Gated Convolution (TDGC), a novel GNN layer for egocentric vision tasks that require a strong *sense of time*;
3) We extend EgoPack to the Moment Queries task, which involves the localization of activities that range from a few seconds to several minutes in duration;
4) Hier-EgoPack achieves strong performance on five Ego4D [8] benchmarks, using the same architecture and showing the importance of cross-task interaction.

II. Related Works

A. Egocentric Vision

Egocentric vision captures human activities from the privileged perspective of the camera wearer, allowing a unique point of view on their actions [9], [10]. Recently, the field has seen rapid development thanks to the release of several large-scale egocentric vision datasets [8], [11], [12], [13], [14], [15]. The rich annotations of these datasets [8], [14] allow to tackle a large number of tasks, including action recognition [16], action anticipation [2], [17], [18], next active object prediction [19], action segmentation [3], [20], episodic memory [21] and long-range temporal reasoning tasks [22], [23], [24]. Previous works in egocentric vision have focused on domain adaptation [25], [26], [27], [28], [29], multi-modal learning [26], [30], [31] and large-scale video-language pretraining [32], [33], [34], [35] to learn better representation for downstream tasks.

B. LLMs for Video Understanding

Large Language Models (LLMs) are becoming increasingly relevant in the video understanding context [36], [37], [38], but suffer from two main limitations, namely the need for ad-hoc task-specific solutions [39], [40] and poor temporal reasoning capabilities [38], [41], [42]. LLaVa [43] is the first approach to combine a pretrained image encoder with a language-only LLM for general image understanding tasks, showing strong zero-shot performance on video understanding tasks [44]. Video-LLaVa [45] leverages distinct image and video encoders and projects them in a common features space, while Video-LLaMa [46] adopts a similar approach for the visual and audio components of the video. Video-ChatGPT [37] is the first multi-modal LLM designed for ChatGPT-like conversations. TimeChat [38] introduces a time-aware frame encoder and a sliding temporal window to process long videos. Recent multi-modal LLMs such as GPT-4o [47] and Qwen2.5-VL [48] have extended their capabilities to video understanding, differing mainly in training data scale and temporal reasoning strategies. LLMs are trained on vast amount of multi-modal inputs, making it difficult to isolate and reuse task-specific knowledge. Hier-EgoPack is designed to explicitly abstract knowledge from different tasks in a format that is easily reusable, as a step towards general video understanding models.

C. Graph Neural Networks for Vision Tasks

Traditional neural networks, including Convolutional Neural Networks (CNNs), have been widely used in computer vision, showing impressive performance on a variety of problems [49], [50], [51]. However, these models often assume data lying on a regular domain, such as images that have a grid-like structure. In recent years, the interest in developing methods able to provide a more general and powerful type of processing has been growing and particular attention has been given to learning methods on graphs. Graph Neural Networks (GNNs) have the innate ability to effectively handle data that lie on irregular domains, such as 3D data [52], [53], robotics [54], molecular chemistry [55], and social or financial networks [56], and to model complex data relations [57]. Recently, transformer-based architectures had a great impact on vision applications. Despite Transformers and GNNs share some similarities in their ability to handle various data types, they are fundamentally different in their core architectures and the specific ways they process

data. GNNs can model the topology of a graph and the relations between nodes while also inheriting all the desirable properties of classic convolutions: locality, hierarchical structures and efficient weights reuse. In video understanding, GNNs have been applied to action localization [20], [58], [59], [60], to build a knowledge graph from human actions [61], to model human-object interactions [62], [63] or to build a topological map of the environment [64].

D. Multi-Task Learning

MTL [65], [66] tackles the problem of learning to solve multiple tasks simultaneously. The development of this strategy is justified by the intuition that complex settings require solving multiple tasks, for instance autonomous driving [67], robotics and natural language processing. Furthermore, these networks can bring the theoretical advantage of sharing complementary information to improve performance. Several works have been done in this direction [4], [67], [68], [69], [70], [71], [72], [73], focusing on which parameters or tasks is better to share [74], [75], [76], [77] and promoting synergies between tasks [78], [79]. Such methods encounter the problem of negative transfer [4] and sharing with unrelated tasks [75], [76] consequently suffering of task competition and not being able to benefit from information sharing between tasks. To overcome these limitations, several methods have been proposed to balance task-related losses [80], [81], [82], to dynamically prioritize tasks [83], to reduce gradient interference between tasks [84] or to exploit task interactions at multiple scales [85]. Unfortunately, all these solutions require extensive task-specific tuning, and are not able to build an holistic perception across tasks.

Few works have explored MTL in egocentric vision [5], [6], [68], [78]. Among these, EgoT2 [5] is the first to investigate semantic affinities among high-level egocentric vision tasks. EgoT2 is designed in two variants, i.e., EgoT2-s and EgoT2-g. EgoT2-s builds a collection of different task-specific models from a set of tasks and learns a transformer-based encoder-decoder network to translate cues from each task that can help the learning process for one of the tasks in the collection. A separate translator is learned for each task in the collection. EgoT2-g extends this architecture to use a shared translator across all the tasks by projecting the tasks in a shared space using a language-based encoding. Hier-EgoPack solves many of the shortcomings of EgoT2, as it is built on a unified architecture for all tasks, making it easy to extend to new tasks, and can support tasks with any temporal granularity.

1) Comparison With EgoPack: EgoPack [6] stands as a fundamentally different paradigm with respect to traditional MTL approaches and is based on a two-stages training pipeline that abstracts the knowledge learned from a set of video understanding tasks in a reusable format. The architecture of EgoPack is built on the intuition that videos can be represented as graphs, where nodes correspond to temporal segments and different tasks can be formulated as distinct operations over this graph structure. This design choice eliminates the need for different architectures across tasks, as in EgoT2. First, a single model is trained on a set of support tasks using a graph-based unified architecture and a set of task-specific heads, one for each task. Then, the knowledge from these support tasks is abstracted in a set of task-specific prototypes. When learning a novel task, the model can peek at the knowledge stored in the prototypes. In this work, we extend EgoPack to support video understanding tasks that span different temporal granularities.

III. Method

We address a cross-task interaction setting, in which an egocentric vision model is trained to reuse previously acquired knowledge from a set of different tasks (*support tasks*) to foster the learning process of any *novel task*. A formal definition of the proposed setting is presented in Section III-A. This work introduces a unified temporal architecture to model tasks with different temporal granularity and strong *sense of time*, i.e., the ability to effectively reason on the order of the events in a video. With this new architecture, we extend EgoPack to tasks that require long range temporal reasoning, e.g., Temporal Action Localization. We call this approach Hier-EgoPack, emphasizing its ability to learn hierarchical video representations that are well suited to various egocentric vision tasks. At the core of Hier-EgoPack is the representation of videos as graphs, following our previous work EgoPack [6], with nodes representing temporal segments of the video and edges reflecting temporal dependencies between them. This design enables the use of a unified graph-based architecture across multiple tasks, facilitating knowledge sharing and allowing tasks to be modeled through distinct graph operations. We present more details on this unified architecture in Section III-B.

A. Setting: Novel Task Learning

A task $\mathcal{T}$ in egocentric vision is defined as a mapping between a video $\mathcal{V}$ and an output space $\mathcal{Y}$. Classification tasks, such as Action Recognition, are defined as a mapping between a video segment $v_i \in \mathcal{V}$ and the corresponding discrete label $y_i \in \mathcal{Y}$. For these tasks, the start and end timestamps of the video segment v_i are known. Differently, the Temporal Action Localization (TAL) task processes the entire video $\mathcal{V}$ and predicts a set of temporally grounded activities, each described by its start and end timestamps and the corresponding action label:

$$\mathcal{T} : \mathcal{V} \rightarrow \{(t_i^s, t_i^e, y_i)\}_i.$$

We streamline the processing for different tasks by feeding the model with untrimmed input videos and aligning the output to the downstream task at a later stage. This alignment process is described more in depth in Section III-C.

The cross-task interaction mechanism of Hier-EgoPack follows a two-stages training process. First, a model is trained on a set of K tasks $\{\mathcal{T}_1, \ldots, \mathcal{T}_K\}$, which we call *support tasks*, in a Multi-Task Learning setting with hard-parameter sharing [86]. The inclusion of multiple tasks in this phase encourages the model to learn more general and task-agnostic representations. Then, the model is presented with a *novel task* $\mathcal{T}_{K+1}$ to learn, without access to the supervision of the *support tasks*. In this scenario, the novel task can benefit from semantic affinities with the previously seen tasks. For example, a model that has

learned to detect object state changes may apply this knowledge for action recognition and vice-versa, as some actions produce object state changes, e.g., *cutting something*, while others do not, e.g., *moving an object*. Our goal is to make these semantic affinities more explicit and exploitable, enabling the novel task to re-purpose these *perspectives* from previous tasks to enhance performance, a necessary step towards more holistic models that seamlessly share knowledge between tasks.

B. A Unified Architecture for Video Understanding

Egocentric vision tasks may provide complementary perspectives but also operate at different temporal granularities, from sub-second interactions to minutes-long activities. To support all these tasks with a unified architecture, we need a model that can perform temporal reasoning hierarchically, progressively integrating fine-grained temporal representations into a broader and more comprehensive understanding. Also, reasoning over long temporal horizons requires the ability to precisely ground and order past and future events. We call this unified architecture the *temporal backbone* of our approach and build it with a task-agnostic design to support any video understanding task with temporal reasoning. The temporal backbone introduced in EgoPack [6] represents a first step in this direction, built on a GNN-based architecture designed to perform temporal reasoning across a set of diverse tasks with similar temporal granularity, limiting its applicability. Indeed, this architecture only partially meets our constraints: while it supports multiple tasks with a shared architecture, it assumes similar temporal granularity across tasks and lacks a robust *sense of time*, as detailed in Section IV-D. Indeed, the SAGE GNN convolutional operator used in EgoPack is invariant to permutations of the input nodes, and temporal ordering of the nodes is only provided by adding a positional encoding to the node embeddings. This strategy is insufficient for tasks that require strong temporal reasoning, as we show in Section IV-D.

We address these challenges by proposing a newly crafted hierarchical GNN-based architecture, specifically designed to support tasks with variable temporal granularity. At the core we place a novel Temporal Distance Gated Convolution (TDGC) layer, able to explicitly encode past and future information, and a temporal sub-sampling operation that progressively computes a coarsened representation of the input video. Starting with high resolution input video features, our architecture progressively aggregates the input on the temporal axis, moving from a local view of the video to a more high-level representation, as shown in Fig. 1. We refer to this architecture as Hier-EgoPack, as it extends EgoPack to deal with different time granularities thanks to its hierarchical processing.

1) Representing Videos as Graphs: A video $\mathcal{V}$ can be seen as a dense sequence of N fixed-length temporal segments encoded as $\mathbf{x} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$, where $\mathbf{x}_i \in \mathbb{R}^D$ represents the features of the corresponding segment v_i computed using a video features extractor $\mathcal{F}$, e.g., EgoVLP [32]. The video can be interpreted as a graph $\mathcal{G}$:

$$\mathcal{G} = (\mathbf{X}, \mathcal{E}, \mathbf{pe}) \tag{1}$$

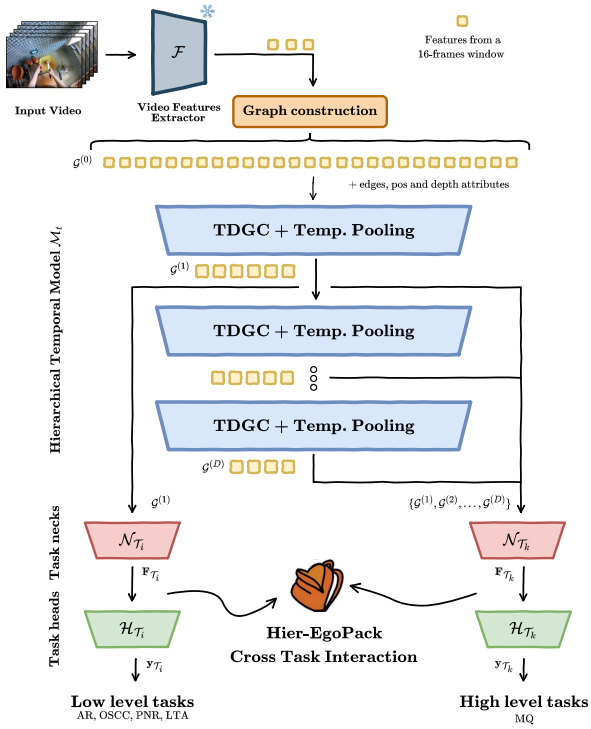

Fig. 1. Overview of the Hier-EgoPack architecture. First, the video is converted into a graph representation $\mathcal{G}^{(0)}$ whose node embeddings are extracted using a frozen video features extractor. The graph is then processed by the *hierarchical temporal backbone* $\mathcal{M}_t$, shared by all the tasks, to progressively learn higher level representations of the input video $\{\mathcal{G}^{(1)}, \mathcal{G}^{(2)}, \ldots, \mathcal{G}^{(L)}\}$. The node embeddings of these graphs are projected by the *task-specific necks* $\mathcal{N}_i$ in the features space of each task $\mathcal{T}_i$ and to the corresponding output space with the *task-specific heads* $\mathcal{H}_i$.

where $\mathbf{X} \in \mathbb{R}^{N \times D}$ is a matrix encoding the features of the graph nodes in its rows, edge $e_{ij} \in \mathcal{E}$ connects nodes i and j with a temporal distance considered relevant when lower than a threshold τ and the attribute $\mathbf{pe} \in \mathbb{R}^N$ encodes the *timestamp* (in seconds). Encoding videos as graphs enables the use of graph neural networks to learn the complex temporal relations between video segments and to cast different egocentric vision tasks as operations on these graphs. The proposed architecture is built on three components:

1) a *temporal backbone* $\mathcal{M}_t$, which uses a stack of TDGC layers and subsampling operations to implement hierarchical temporal reasoning;
2) a set of *task-specific projection necks* $\mathcal{N}_k$ mapping the node embeddings to the features space of task $\mathcal{T}_k$;
3) a set of *task-specific heads* $\mathcal{H}_k$ that map to the output space of each task.

Let $\mathcal{G}^{(0)}$ represent the initial graph of the input video $\mathcal{V}$, where each node's position $\mathbf{pe}$ is initialized to the midpoint of the corresponding video segment. At each stage l, the *temporal backbone* $\mathcal{M}_t$ performs temporal aggregation on the input graph $\mathcal{G}^{(l)}$ and outputs an updated graph $\mathcal{G}^{(l+1)}$. This is done using a sequence of TDGC layers and temporal subsampling operations to progressively enlarge the temporal extent of the nodes while reducing the nodes cardinality of the graph. Subsampling is implemented as a mean/max pooling operation over each node and its neighbors, then removing every alternate node, halving

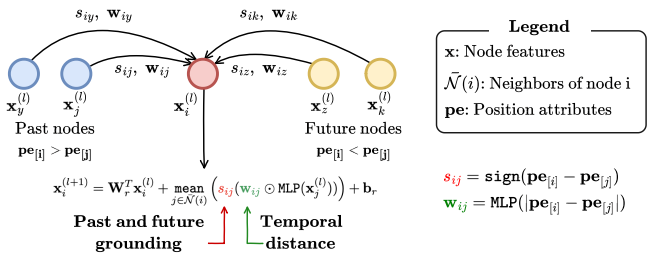

Fig. 2. Temporal Distance Gated Convolution layer (TDGC), specifically designed to integrate *past and future events grounding* (s_{ij}) and to *reason about the temporal distance* between nodes ($\mathbf{w}_{ij}$) in the aggregation step.

the total number of nodes. The edges of the graph are recomputed accordingly by scaling the position of each node by a factor 2^l, where l is the index of the stage of the hierarchical temporal backbone. Overall, the output of the temporal backbone $\mathcal{M}_t$ maps the input graph $\mathcal{G}^{(0)}$ to a set of graphs:

$$\mathcal{M}_t : \mathcal{G}^{(0)} \rightarrow \{\mathcal{G}^{(1)}, \mathcal{G}^{(2)}, \ldots, \mathcal{G}^{(L)}\}, \tag{2}$$

where L is the total number of stages in the backbone and each graph $\mathcal{G}^{(\cdot)}$ is a progressively coarsened representation of the input video. The number of stages L depends on the task: for fine-grained tasks, e.g., AR or OSCC, a single stage is enough, while we use multiple stages for tasks that reason over a longer horizon. More details are reported in Section IV-B. The architecture of the *temporal backbone* is shown in Fig. 1.

2) Temporal Distance Gated Convolution (TDGC): Each stage of the *temporal backbone* $\mathcal{M}_t$ is built as a stack of N_l GNN layers, which we call Temporal Distance Gated Convolution (TDGC). These layers are designed to preserve and encode the temporal sequence of information, capturing the relative past and future dependencies between nodes. The proposed graph convolution layer, visualized in Fig. 2, is explicitly designed to incorporate the relative positions between the root node and its neighbors in the message passing step. More specifically, given two nodes i and j at layer l, we compute s_{ij} as the sign of the relative temporal distance between the nodes and $\mathbf{w}_{ij}$ as a learnable projection of their relative distance (in absolute value):

$$s_{ij} = \texttt{sign}(\mathbf{pe}^{(l)}_{[i]} - \mathbf{pe}^{(l)}_{[j]}), \quad \mathbf{w}_{ij} = \texttt{MLP}(|\mathbf{pe}^{(l)}_{[i]} - \mathbf{pe}^{(l)}_{[j]}|). \tag{3}$$

These two factors are used to re-weight the contribution of each node j in the aggregation step, as follows:

$$\mathbf{x}'_j = \texttt{MLP}\left(\mathbf{x}^{(l)}_j\right) = \phi(\mathbf{W}^T_n \mathbf{x}^{(l)}_j + \mathbf{b}_n), \tag{4}$$

$$\mathbf{x}^{(l+1)}_i = \mathbf{W}^T_r \mathbf{x}^{(l)}_i + \underset{j \in \bar{\mathcal{N}}(i)}{\texttt{mean}} \left(s_{ij}(\mathbf{w}_{ij} \odot \mathbf{x}'_j)\right) + \mathbf{b}_r, \tag{5}$$

where $\mathbf{x}^{(l)}_i$ are the features of the node i at layer l, $\bar{\mathcal{N}}(i)$ is the set of neighbors of node i, $\mathbf{W}_n$, $\mathbf{W}_r$ and $\mathbf{b}_n$, $\mathbf{b}_r$ are learnable weights and biases respectively. Subscript r refers to the contribution of the root node. Our TDGC layer is inspired by previous works on Temporal Action Localization which used 1D temporal convolution [3], [87]. However, unlike common 1D convolutions, TDGC employs shared weights to aggregate past and future nodes, enabling its application to video segments of arbitrary length and to graphs in which the relative temporal distance between nodes is not fixed.

C. Task-Specific Components

The temporal backbone $\mathcal{M}_t$ is shared between all downstream tasks and is designed to support task-agnostic temporal reasoning over a stream of fixed-length video segments. After the backbone, we attach a separate neck $\mathcal{N}_k$ for each task $\mathcal{T}_k$ to project the node embeddings into the feature space of the corresponding task and possibly aligning them to the temporal boundaries of the task. Features $\mathbf{X}^{(l)}$ from the temporal backbone are first projected with the task neck $\mathcal{N}_k$, implemented as a two-layers MLP, to obtain $\mathbf{X}^{(l)}_k$:

$$\mathbf{X}^{(l)}_k = \mathcal{N}_k\left(\mathbf{X}^{(l)}\right) \quad \text{with} \quad \mathcal{N}_k : \mathbb{R}^D \rightarrow \mathbb{R}^D. \tag{6}$$

The neck is shared for all the output graphs of the *temporal backbone*. Then, for tasks defined on input segments with known temporal boundaries, e.g., Action Recognition, we align the node embeddings with the task annotations. For each video segment $v_i \in \mathcal{V}$ annotated for the task $\mathcal{T}_k$, we aggregate the node embeddings that are between the start s_i and end e_i boundaries of the segment to obtain $\mathbf{F}^{(l)}_{k,[i]}$:

$$\mathbf{F}^{(l)}_{k,[i]} = \texttt{align}(\mathbf{X}^{(l)}_k, s_i, e_i) = \underset{j:\ s_i < \mathbf{p}^{(l)}_{[j]} < e_i}{\texttt{mean}} \mathbf{X}^{(l)}_{k,[j]}, \tag{7}$$

where i and j are row-indices and $\mathbf{F}^{(l)}_{k,[i]}$ are the task-specific features of segment v_i of the video for task $\mathcal{T}_k$. Other tasks, e.g., Temporal Action Localization, operate on the full video and do not require task-specific alignment. In such case, the task-specific features $\mathbf{F}^{(l)}_k$ are set equal to the output of the task-specific neck $\mathbf{X}^{(l)}_k$.

D. Building a Backpack of Reusable Skills

To solve the *novel task* $\mathcal{T}_{K+1}$, the naive approach would be to finetune the model, adding new task-specific neck $\mathcal{N}_{K+1}$ and head $\mathcal{H}_{K+1}$ and possibly updating the *temporal backbone* $\mathcal{M}_t$. However, finetuning may not fully leverage the insights from other tasks as it could result in the loss of the previously acquired knowledge, as the model adapts to the new task. Instead, we explicitly model the perspectives of the *support tasks*, i.e., the set of tasks the model has learned in the MTL pre-training step, as a set of task-specific prototypes that can be accessed by the novel task. This approach was originally proposed as part of EgoPack [6] and we provide an overview in Fig. 3. We collect these task-specific prototypes from videos annotated for action recognition, as human actions can be seen as the common thread behind the different tasks.

Practically, we forward these action samples through the temporal backbone, align them based on the action recognition annotations and project their features using the task-specific necks $\mathcal{N}_k$ of each task to obtain the task-specific features $\mathbf{F}_k$ for each task in the MTL pre-training phase. Each row in $\mathbf{F}_k$ encodes the perspective of each task for the same video segment. To summarize these features into prototypes we aggregate

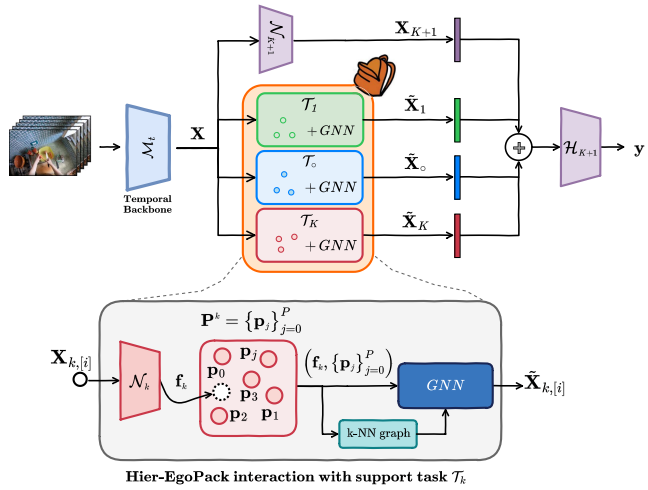

Fig. 3. Learning a novel task with a backpack. After the Multi-Task training phase, we extract a set of prototypes $\mathbf{P}^k$ that summarize what the network has learned from each *support task* $\mathcal{T}_k$, like a backpack of skills that we can carry over. In this *Cross-Tasks Interaction* phase, the network can peek at these different task-perspective to enrich the learning of the novel task.

them according to the action label of the corresponding action segment, i.e., a *verb* and *noun* pair:

$$\mathbf{P}^k = \{\mathbf{p}_0^k, \mathbf{p}_1^k, \dots, \mathbf{p}_P^k\} \in \mathbb{R}^{P \times D}, \tag{8}$$

for each task $\mathcal{T}_k$, where P is the number of unique *(verb, noun)* pairs in the dataset and D is the size of the task-specific features. These prototypes are frozen and represent a *summary* of what the models has learned during the multi-task pre-training process, creating an abstraction of the gained knowledge. They can be then reused when learning a *novel task*, like a backpack of skills that the model can carry over. Notably, storing the model's knowledge in the prototypes allows for fine-tuning the temporal backbone, which is especially valuable when the novel task has a different temporal granularity compared to the previous tasks.

E. Learning a Novel Task With a Backpack

Let us now consider the case in which we want to solve a novel task $\mathcal{T}_{K+1}$. The model can exploit the perspective of the previously seen tasks by comparing the output of the task-specific necks for tasks $\mathcal{T}_{1,\dots,K}$ with their corresponding prototypes. When learning the novel task $\mathcal{T}_{K+1}$, the output graphs of the *temporal backbone* are forwarded through all projection necks to obtain the task-specific features $\mathbf{X}_k^{(l)}$, as defined in (6). To improve readability we hereinafter omit the superscript indicating the specific stage l at the temporal backbone stage. These features are used as *queries* to match the corresponding task prototypes $\mathbf{P}^k$, using k-NN in the features space to look for the closest prototypes. Task features and their neighboring prototypes form a *graph-like* structure, on which message passing is performed to enrich the task-specific features $\mathbf{X}_k^{(l)}$, following an iterative refinement approach, using M layers of SAGE convolution. At each layer m of Hier-EgoPack, we update the features $\mathbf{X}_{k,[i]}$ from stage l of the temporal backbone by combining them with its closest prototypes $\bar{\mathcal{N}}(i)$:

$$\mathbf{X}_{k,[i]}^{(m+1)} = \mathbf{W}_r^{(m)} \mathbf{X}_{k,[i]}^{(m)} + \mathbf{W}^{(m)} \cdot \underset{\mathbf{p}_j^k \in \bar{\mathcal{N}}(i)}{\texttt{mean}} \mathbf{p}_j^k, \tag{9}$$

where $\mathbf{p}_j^k \in \bar{\mathcal{N}}(i)$ are the *activated prototypes* for the given task, i.e., the set of closest task-specific prototypes in $\mathbf{P}^k$ with respect to $\mathbf{X}_{k,[i]}$, and $\mathbf{W}_r^{(m)}$, $\mathbf{W}^{(m)}$ are learnable projections of the input features and the aggregated neighbors, respectively. Eq. (9) is applied to features from all l stages of the hierarchical temporal backbone. Notably, only the task features are refined while the task prototypes remain frozen to preserve the original perspectives seen by the network. We denote the output of this interaction process as $\tilde{\mathbf{X}}_k^{(l)}$. These features are then possibly aligned to the boundaries of the novel task to obtain $\tilde{\mathbf{F}}_k^{(l)}$, as discussed in Section III-C.

In this process, the *task-specific necks* of the support tasks $\mathcal{N}_{1,\dots,K}$ are initialized from the multi-task training and updated during the task-specific finetuning process, allowing the model to explore the set of task prototypes and to select the most informative ones for each input sample. Moreover, to allow the model to learn complementary cues specific to the novel task, we add a new pair of neck $\mathcal{N}_{K+1}$ and head $\mathcal{H}_{K+1}$. We evaluate different fusion strategies to integrate the novel task with the perspectives gained from the previous tasks. In *features-level* fusion, we average the task-specific features for the novel task $\mathbf{F}_{K+1}$ with the *refined* perspectives from the previous tasks $\tilde{\mathbf{F}}_k$. In *logits-level* fusion, we keep a set of separate heads, one for each task $\mathcal{T}_{1,\dots,K}$, feed the features $\tilde{\mathbf{F}}_k$ to each head separately and sum their outputs, as in the original EgoPack implementation. Intuitively, this approach allows each task to cast a vote on the final prediction, based on its perspective on the same video segment.

F. Training Process

We train our models using only supervision of the known task, for both single and multi-task models. More details are reported in Section IV-B. When training Hier-EgoPack, we finetune the *temporal backbone*, the task-specific projection necks and the heads. Gradient updates from the support tasks are not propagated to the *temporal backbone*.

IV. Experiments

We first introduce in Section IV-A the tasks addressed in this work and the implementation details for our models and the *Task-Translation* baseline in Section IV-B. We report quantitative results for Hier-EgoPack in Section IV-C, evaluate different design choices in Section IV-D and demonstrate the effectiveness of our approach on the test-set in Section IV-E. Finally, in Section IV-F we show qualitative results demonstrating the interaction process of Hier-EgoPack.

A. Setting

We validate our approach on Ego4D [8], a large scale dataset with 3.6k hours of egocentric videos capturing unscripted daily-life human activities, focusing on five Ego4D benchmarks that cover different temporal granularities. *Fine-grained tasks* focus

on short-term understanding of the video, usually a few seconds long, and include:

- *Action Recognition (AR):* given a video segment, predict the verb and noun action labels describing the interaction from a taxonomy of 115 and 478 verb and noun classes respectively. We report verb and noun top-1 accuracy.[1]
- *Object State Change Classification (OSCC):* given a video segment, predict the presence (or absence) of an object state change, e.g., a glass being filled (transition from *empty* to *full*). We report accuracy.
- *Point of No Return (PNR):* given a video segment containing an object state change, predict the temporal frame when the change happens. Predictions are evaluated using the absolute temporal distance from the ground truth.
- *Long Term Anticipation (LTA):* given a video segment, predict the sequence of Z future actions (verb and noun label pairs) the camera wearer is likely to perform next. Performance is measured in terms of verbs and nouns Edit Distance (ED) between the predicted sequence and the ground truth, for the best sequence out of K predictions. In Ego4D, $Z = 20$ and $K = 5$.

Other tasks may require both short and long term understanding of the input video. Among these, we analyze the *Moment Queries (MQ)* task, which requires predicting the set of activities performed in the video among 110 labels with the corresponding start and end timestamps. For all tasks, we use the version *v1* of the annotations.

B. Implementation Details

Hier-EgoPack is built using pre-extracted features from fixed-size video segments. In all experiments the backbone used for feature extraction is kept frozen. We use EgoVLP features pretrained on EgoClip [32] and extracted using a window of 16 consecutive frames with an equivalent stride. EgoVLP features have size 256. For comparison with EgoPack in Table II, we use Omnivore Video Swin-L [88] features pre-trained on Kinetics-400 [89], released as part of Ego4D [8] and extracted using dense sampling over a window of 32 frames with a stride of 16 frames and features size 1536. In principle, Hier-EgoPack is agnostic to the features extractor and could adopt other architectures. We train all the single, multi-task and Hier-EgoPack models for 15 epochs, using the Adam optimizer. Learning rate is set to $1e{-}4$ for all tasks, with the exception of the OSCC and PNR tasks which use $1e{-}5$, and follow a cosine annealing schedule with a linear warm-up of 5 epochs. We repeat our experiments three times with different random seed and report the average performance. All tasks share the same temporal and cross-task interaction architecture, with minimal task-specific hyper-parameter tuning. The task prototypes are built using samples from the train split of the AR task.

1) Task-Specific Design Choices: EgoPack constructs the input graph differently based on the task, i.e., each action or sub-segment is mapped to a different node in AR or OSCC respectively, which may result in inconsistencies in how segments with different temporal granularities are processed by the temporal backbone. On the contrary, with Hier-EgoPack we standardize the graph construction process for all tasks. Specifically, features from fixed-length segments are extracted densely from the entire video and each segment is mapped to a node of the graph. Temporal reasoning is performed on these *dense* temporal graphs, followed by a task-specific projection $\mathcal{N}_k$ and an optional alignment step. Depending on the temporal granularity of the downstream task, we take the output processed graphs of the temporal model after the first stage $\mathcal{G}^{(1)}$ (*fine-grained tasks*) or from all the stages $\{\mathcal{G}^{(1)}, \mathcal{G}^{(2)}, \ldots, \mathcal{G}^{(L)}\}$ (*variable-resolution tasks*). For tasks in which temporal boundaries are known, features within the boundaries are averaged to obtain a single instance-level embedding as input for the task-specific neck. With the exception of MQ, all tasks fall into this category. For MQ, we predict an action for each segment in the input video and use Non-Maximum Suppression (NMS) to filter predictions, consistently with previous approaches. For NMS, we use the same configuration as ActionFormer [3] and set the σ parameter to 2.0, which was empirically found to reduce the penalty on *near-replicate* predictions [90]. Therefore, no specific alignment is needed for this task.

Task-specific necks are implemented as two-layers MLPs. The heads are also implemented as multi-layer projections that map to the output space of the task, with the exception of the LTA task. In this case, we first build *on-the-fly* a graph with K nodes initialized to the output of the temporal model, where K is the number of future actions to predict. We then process this graph with a two layers TDGC, before feeding the features to the verb and noun classifiers.

AR, OSCC and LTA are trained with standard cross entropy loss, while PNR uses binary cross entropy. The classification and regression heads of the MQ task are trained with the focal [91] and DIoU [92] losses respectively, following the same protocol as ActionFormer [3] to match predictions at different scales with their temporally closest ground truth.

2) Task-Translation Baseline Implementation: Due to the differences in the network architecture and training data employed, a comparison between Hier-EgoPack and EgoT2 [5] is not straightforward. Indeed, EgoT2's Single Task are based on SlowFast [93] for AR and LTA, I3D ResNet-50 [89] for OSCC and PNR and VSGN [87] for MQ. These models are end-to-end trained on the benchmarks' data, unlike Hier-EgoPack which relies on pre-extracted features and does not train the video feature extractor. Therefore, we introduce a comparable baseline, which we call *Task Translation*, by adapting the cross-task translation mechanism of EgoT2 to our setting. As in EgoT2s, *Task Translation* learns a transformer encoder on top of the Single Task models to combine the perspective of the different tasks. Furthermore, EgoT2 supports only tasks with homogeneous temporal granularity. With *Task Translation*, we extend the translation mechanism to support tasks with different temporal granularities and include in this analysis the same tasks as Hier-EgoPack.

Formally, *Task Translation* combines a set of K Single Task models trained independently. Each Single Task model outputs

[1] This task is not an official Ego4D [8] task and was initially introduced by EgoT2 [5] using the LTA annotations.

TABLE I
HIER-EGOPACK ON EGO4D HUMAN-OBJECT INTERACTION (HOI) AND MOMENT QUERIES (MQ) TASKS

	AR		OSCC	LTA		PNR	MQ
	Verbs Top-1 (%)	**Nouns Top-1 (%)**	**Acc. (%)**	**Verbs ED (↓)**	**Nouns ED (↓)**	**Loc. Err. (↓)**	**mAP**
Ego4D Baselines [8]	22.18	21.55	68.22	0.746	0.789	0.62	6.03
EgoT2s [5]	23.04	23.28	72.69	0.731	0.769	**0.61**	N/A
EgoPack [6]	25.10	31.10	71.83	0.728	0.752	**0.61**	N/A
Single Task	26.93	33.50	75.22	0.728	0.752	0.62	20.2
MTL	26.31	33.90	74.79	0.730	0.754	0.62	18.5
MTL + FT	26.71	33.51	75.00	0.728	0.749	**0.61**	19.9
MTL + HT	26.07	33.20	74.27	0.729	0.748	0.62	N/A
Task-Translation†	26.10	33.83	**76.42**	0.729	0.750	0.63	20.5
Hier-EgoPack	**27.30**	**34.65**	75.60	**0.725**	**0.741**	**0.61**	**21.0**

Single Task uses the same hierarchical GNN-based architecture to model all tasks, with minimal task-specific differences. *Multi-Task Learning (MTL)* uses hard parameter sharing to jointly learn all tasks, which may result in negative transfers. *Ego-T2s* [5] learns to translate features across tasks to optimize the primary task. *Hier-EgoPack* builds on the unified architecture of the Temporal Graph and learns to exploit the perspective of different tasks for efficient knowledge transfer to the novel task. Performances of Hier-EgoPack are evaluated over three runs with different random seeds using accuracy for AR and OSCC, Edit Distance for LTA, temporal localization error (in seconds) for PNR and mAP for MQ. †*Task-Translation* implements the same cross-task translation mechanism of EgoT2s [5] using a frozen EgoVLP backbone, as for Hier-EgoPack. Best results are reported in bold, second best are underlined.

a sequence of N_k task-specific tokens $\mathbf{F}_k = [\mathbf{f}_k^1, \mathbf{f}_k^2, \ldots, \mathbf{f}_k^{N_k}]$ with $\mathbf{f}_k^i \in \mathbb{R}^D$, along with the position attribute $\mathbf{pe}_k \in \mathbb{R}^{N_k}$, as defined in Section III. Task-specific tokens and the position attribute are concatenated on the sequence dimension to obtain the full features $\mathbf{F} \in \mathbb{R}^{N \times D}$ and position attribute $\mathbf{pe} \in \mathbb{R}^N$, where N is the total number of tokens across all the tasks. We define the *Task Translation* operation as $\tilde{\mathbf{F}} = \texttt{ENC}(\mathbf{F}, \mathbf{A})$, where $\mathbf{A}$ is a binary attention mask defined as:

$$\mathbf{A}_{[ij]} = \begin{cases} 1 & \text{if } \left|\mathbf{pe}_{[i]} - \mathbf{pe}_{[j]}\right| \leq 2^l \\ 0 & \text{otherwise} \end{cases}, \tag{10}$$

where l is the index of the stage in the hierarchical backbone that produced features $\mathbf{f}_i$. The mask restricts the self-attention operation to tokens that are within the same temporal window. We parameterize ENC as a transformer encoder with l layers and h attention heads, with the same output size as the input features. Finally, we take the slice of the transformer output $\tilde{\mathbf{F}}$ corresponding to the features of the primary task and forward them through the task-specific head.

C. Quantitative Results

We show the main results of Hier-EgoPack in Table I, comparing our approach with the Ego4D baselines [8], the task-translation framework EgoT2 [5] and the previous iteration of our work EgoPack [6]. We also compare Hier-EgoPack with recent LLMs in the Supplementary Materials.

We proceed incrementally from the *Single Task* models, i.e., each task is trained separately using our GNN-based hierarchical architecture. Conversely, Ego4D baselines and EgoT2 use SlowFast [93] for AR and LTA, I3D ResNet-50 [89] for OSCC and PNR and VSGN [87] for MQ, with different configuration and hyper-parameters for each task. In contrast, our *Single Task* models employ the same architecture and a pair of task-specific neck and head. *Multi-Task Learning* (MTL) baselines are built following the same approach, i.e., sharing the same architecture across all the tasks. In this setting, we observe suboptimal performance in some tasks, particularly in AR (Verb), OSCC, and MQ. We attribute this to potential negative transfer effects.

TABLE II
COMPARISON OF EGOPACK AND HIER-EGOPACK USING OMNIVORE FEATURES

	AR		OSCC	LTA		PNR
	Verbs	**Nouns**	**Acc.**	**Verbs**	**Nouns**	**Err.**
Single Task [6]	24.25	30.43	71.26	0.754	0.752	**0.61**
MTL [6]	22.05	29.44	71.10	0.740	0.746	0.62
EgoPack [6]	25.10	31.10	**71.83**	**0.728**	0.752	**0.61**
Single Task	24.41	31.41	71.74	0.733	0.743	**0.61**
MTL	23.72	31.43	71.33	0.737	0.756	0.62
Hier-EgoPack	**25.33**	**31.64**	71.77	0.729	**0.741**	**0.61**

Comparison between *EgoPack* and *Hier-EgoPack* using the same input features (Omnivore) and tasks, i.e., AR, OSCC, LTA and PNR.

We also consider a *MTL+FT* baseline in which the MTL model is finetuned on the novel task, and *MTL+HT* which takes the frozen temporal backbone from the MTL training and learns new task-specific neck $\mathcal{N}_K$ and head $\mathcal{H}_K$ for the novel task. These baselines exhibit comparable performance to the *Single Task* models, showing that fine-tuning multi-task models is not the ideal approach to transfer knowledge across tasks as it does not explicitly exploit the semantic similarities and perspectives offered by different tasks.

1) Task-Translation Baseline Results: Task-Translation shows consistent improvements compared to both Single Task and Multi-Task models, with the sole exception of AR. These results prove the effectiveness of the cross-task translation mechanism and show that different tasks learn representations that are partially complimentary to each other. However, we remark the *Task-Translation* mechanism is inefficient by design as it requires different models for each supported task. Each single task model in the ensemble looks at a different perspective for the same input, without explicitly recalling the entire knowledge gained by the models. In contrast, the task prototypes in Hier-EgoPack provide a comprehensive and easy-to-access abstraction of the model's learned knowledge, enabling the extraction of relevant insights tailored to the specific sample and task.

2) Comparison With EgoPack: We compare Hier-EgoPack with our previous iteration EgoPack [6] in Table II, using the

TABLE III
ABLATIONS ON DIFFERENT COMPONENTS OF THE HIERARCHICAL TEMPORAL MODEL

N_l	mAP	R@1	R@5
1	18.57	31.47	53.36
2	**20.21**	34.15	56.78
3	19.75	35.08	57.23
4	19.93	**35.24**	**59.28**

Number of TDGC layers in each stage of the temporal backbone.

Pooling	mAP	R@1	R@5
batch ss.	19.95	34.30	57.69
video ss.	19.35	33.43	56.53
max	19.87	**34.41**	**58.14**
mean	**20.21**	34.15	56.78

Pooling strategy to progressively reduce the number of nodes in the temporal backbone.

Temp. Thresh. τ	mAP	R@1	R@5
1	18.21	31.09	54.92
2	**20.21**	**34.15**	**56.78**
4	19.63	32.49	54.53
8	20.07	31.67	52.46

Temporal distance threshold to define a connection between nodes in the temporal graph.

TABLE IV
ABLATIONS ON DIFFERENT GNNS FOR THE HIERARCHICAL BACKBONE

GNN	mAP @ IoU 0.1	0.2	0.3	0.4	0.5	Avg
Permutation-Invariant layers						
GCN [95]	21.43	18.46	15.51	12.16	9.21	15.35
GAT [96]	21.58	18.57	15.58	12.12	9.12	15.39
SAGE [97]	21.95	19.00	15.99	12.49	9.21	15.73
Temporal-aware layers						
SAGE + PE.† [97]	25.22	21.38	17.82	13.61	10.28	17.66
SGCN [98]	25.35	22.32	19.58	17.03	14.39	19.73
TDGC (w/o s_{ij})	21.27	18.10	15.35	12.09	8.79	15.12
TDGC (w/o $\mathbf{w}_{ij}$)	24.98	21.99	19.55	16.85	14.25	19.52
TDGC	**25.83**	**22.93**	**20.17**	**17.38**	**14.73**	**20.21**

Comparison between TDGC and other GNN layers for the stages of the temporal backbone on the MQ task. † A sinusoidal positional encoding is added to the nodes of the input graph.

same *fine-grained* tasks, i.e., AR, OSCC, LTA and PNR, and same pre-extracted features (Omnivore). Hier-EgoPack, thanks to its novel GNN layer with strong temporal reasoning, has on average better performance compared to the Single Task models from the original EgoPack.

D. Ablations

We evaluate different design choices for the hierarchical temporal backbone in Tables III and IV, focusing on the Moment Queries (MQ) task which requires temporal reasoning at multiple granularities, thus exploiting the hierarchical architecture in its entirety.

Number of GNN layers: The number of convolutional layers at each stage has a mild impact on performance, as it tends to saturate after two layers (Table III-left). Increasing the number of layers expands the receptive field at each stage, a goal already achieved by our pooling and hierarchical aggregation steps. Consequently, adding more layers appears redundant given the model's hierarchical structure.

Pooling strategy: We evaluate different approaches to *reduce the temporal resolution of graph nodes* between subsequent layers of the temporal model (Table III-middle). The *batch* strategy selects alternate nodes from the batch, without considering video boundaries, which results in some noise in the node selection process. Differently, *video* selects alternate nodes from each video separately. The *mean* and *max* strategies pool features from all the neighbors of each node, corresponding to past and future segments. On the MQ task, we observe a noticeable gap between the first two strategies that drop half the nodes and the *mean* and *max* strategies which operate on the neighbors of each node and can better forward task-relevant information to the next layers.

Temporal threshold: The τ parameter controls the number of neighbors at each node in the temporal graph, as we consider the existence of an edge e_{ij} between two nodes i and j only if their relative temporal distance is less than the threshold τ. We observe that small values of τ are sufficient and performance deteriorates quickly with larger values, especially in terms of recall (Table III-right). Also, the use of a smaller neighborhood is compensated by the hierarchical nature of our temporal backbone.

GNN layer: In Table IV, we analyze the impact of different GNN layers on MQ performance. At each layer, the neighborhood of a node is the set of nodes within a fixed relative distance τ from the root node, which makes the GNN operate on local temporal segments of the video. We evaluate two different approaches: i) permutation invariant (PI) layers, which ignore the local temporal ordering of the nodes in the neighborhood, and ii) layers that explicitly incorporate temporal grounding, i.e., node ordering, into their processing. Both strategies achieve reasonable performance. However, the absence of temporal ordering in the approaches from the first group prevents them from properly aggregating past and future nodes, resulting in subpar performance compared to strategies that include temporal grounding.

We evaluate different strategies to add temporal grounding to the GNN layers. The simplest approach, *SAGE + PE*, adds an absolute positional encoding to the node embeddings of the input graph. This method, already used by EgoPack, works well in tasks that do not require strong temporal reasoning. Despite its simplicity, it outperforms all PI approaches, underscoring the significance of precise node ordering for more *temporal-aware* tasks, such as the MQ. A more advanced strategy is SGCN [97], which extends GCN by using different projections for the node embeddings corresponding to past and future segments in the neighborhood. To design an effective GNN layer for diverse video understanding tasks, we focus on two key temporal reasoning principles: (i) the ability to distinguish between past and future nodes in the aggregation phase and (ii) the relevance of each node should depend on its relative temporal distance. SGCN addresses the first point but does not consider the relative temporal distance of the nodes, giving the same importance to close and distant nodes. Also, past and future node embeddings are projected differently despite possibly encoding the same event. Our intuition is that the relative temporal distance should not affect the semantic content of the nodes, and therefore their

TABLE V
TEST-SET RESULTS FOR MOMENT QUERIES (MQ)

Method	Features	Validation mAP @ IoU 0.1	0.3	0.5	Avg	Test mAP Avg
Ego4D Baseline [8]	SlowFast	9.10	5.76	3.41	6.03	5.68
VSGN [87]	EgoVLP	16.6	11.5	6.57	11.4	10.3
ActionFormer† [3]	EgoVLP	26.8	20.6	14.5	20.6	17.5
ASL‡ [99]	EgoVLP	**29.5**	**23.0**	**16.1**	**22.8**	**22.3**
Hier-EgoPack	EgoVLP	27.0	21.0	15.2	21.0	18.0

We report mAP at different thresholds and average mAP in [0.1:0.1:0.5] on the validation and test sets of Moment Queries (MQ). Best results in bold, second best underlined. † Reproduced results on the test set (not present in the original paper).‡ ASL [99] is a considerably larger model (350.7 MParams) compared to Hier-EgoPack (37.1 MParams). Also, models are trained on both train and validation splits and three different models are ensembled at test-time for better performance [100].

TABLE VI
TEST-SET RESULTS FOR LONG TERM ANTICIPATION (LTA)

Method	Version	Verb ED	Noun ED	Action ED
Vision-based approaches				
SlowFast [8]	*v1*	0.739	0.780	0.943
EgoT2 [5]	*v1*	0.722	0.764	0.935
HierVL [34]	*v1*	0.724	0.735	0.928
I-CVAE [101]	*v1*	0.741	0.740	0.930
EgoPack [6]	*v1*	**0.721**	0.735	0.925
Hier-EgoPack	*v1*	0.726	**0.716**	**0.924**
LLM-based approaches				
AntGPT [39]	*v1*	0.658	0.655	0.881
PALM [102]	*v1*	**0.656**	**0.640**	**0.861**

We report Verb, Noun and Action Edit Distance on the test set of Long Term Anticipation (LTA), separately for *vision-based* and *LLM-based* approaches.

projection, but only how nodes are combined in the aggregation phase. By using our TDGC layer we adopt the same projection for all nodes and encode the temporal distance between the nodes in the aggregation step.

To analyze the impact of the aforementioned key temporal reasoning principles, Table IV presents an ablation study on the design choices for our TDGC. The results clearly show a significant performance drop when the s_{ij} coefficients are removed, as this prevents distinguishing between past and future nodes during aggregation. Similarly, omitting the relative position attributes $\mathbf{w}_{ij}$, which differentiates between temporally close and distant nodes, results in suboptimal performance in the MQ task.

E. Benchmarks

We compare Hier-EgoPack on the test set of MQ and LTA benchmarks, to validate the improvements and soundness of our approach. In this setting, a fair comparison between methods is challenging because of the use of different backbones or feature extractors, supervision levels, ensemble strategies, and challenge-specific tuning, such as training also on the validation set.

Moment Queries (MQ): We compare different approaches using EgoVLP features and with the official Ego4D baseline in Table V. VSGN [87] is a two-stages method featuring a pyramid network to exploit cross-scale correlations in the input video. ActionFormer [3] is a single-stage method that combines a multi-scale transformer encoder with a lightweight convolutional decoder. ASL [98] extends ActionFormer by reweighting the predictions based on their distance from the corresponding ground truth segment. ASL is a much larger model in terms of trainable parameters than Hier-EgoPack (350.7 versus 37.1 MParams) and the test-set results are obtained with an ensemble of three models, each trained with different hyperparameters on the combination of the training and validation splits. We include this model in our analysis because of its relevance and use of EgoVLP features, although it is not directly comparable with the other approaches.

In particular, Hier-EgoPack significantly outperforms VSGN and ActionFormer, despite having a generic architecture not specifically designed for the task.

Long Term Anticipation (LTA): We compare different approaches for the LTA task in Table VI. In particular, we distinguish between *vision-based* and *LLM-based* approaches, with the former relying only on visual reasoning and the latter integrating LLMs into their pipeline. Hier-EgoPack achieves SOTA performance on the *noun* and *action* metrics in the *vision-based* category, with similar performance compared to EgoPack on the *verb* metric.

F. Qualitative Results

In this section, we analyze how Hier-EgoPack leverages knowledge abstractions from the *support tasks* (collected in the form of prototypes) to aid the learning of a *novel task*. Specifically, we visualize the *activated prototypes* (i.e., the set of prototypes each *support task* looks at) during the interaction process of Hier-EgoPack across different novel tasks and quantify task activation consensus, a measure of the complementarity among support tasks in aiding the learning of a novel task.

Prototypes activations: We show in Fig. 4 the activation frequency for the task-specific prototypes for a subset of *novel tasks*, considering the Top-20 most activated prototypes. Due to the large number of prototypes, we aggregate them based on their verb labels to enhance the readability of the plots. Some tasks, i.e., OSCC and LTA, also show more similar activations frequencies for the prototypes corresponding to the same label while Moment Queries have a much larger variability in prototypes activations.

Activations consensus: The goal of this analysis is to showcase how a *novel task* can leverage the perspectives from a set of *support tasks*, reusing previously learned knowledge stored in the form of prototypes. To this end, we expect Hier-EgoPack to extract complementary cues from each *support task*. We define the *activations consensus* as the degree to which different tasks activate prototypes corresponding to the same label for a given sample of the *novel task*. A low consensus suggests that the support tasks capture more diverse cues, i.e., different tasks activate different prototypes, whereas a high consensus indicates that activations are more coherent across tasks. We show in Fig. 5 the average activation consensus for different novel tasks. Fine-grained tasks, e.g., AR, LTA and OSCC, have higher average consensus compared to MQ. We attribute this

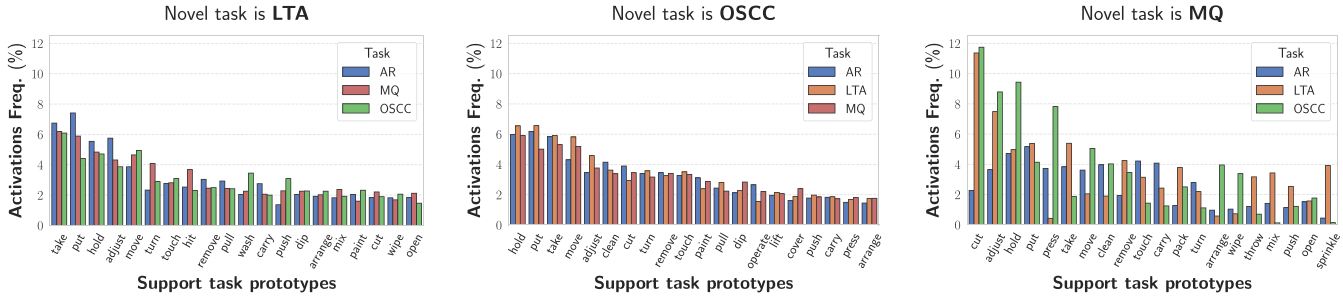

Fig. 4. Activation frequency for the task-specific prototypes from different ***support tasks***. We focus on the Top-20 most activated prototypes across the *support tasks*. LTA and OSCC have more uniform activations across different support tasks, i.e., they look at similar prototypes, while MQ exhibit more diverse activations.

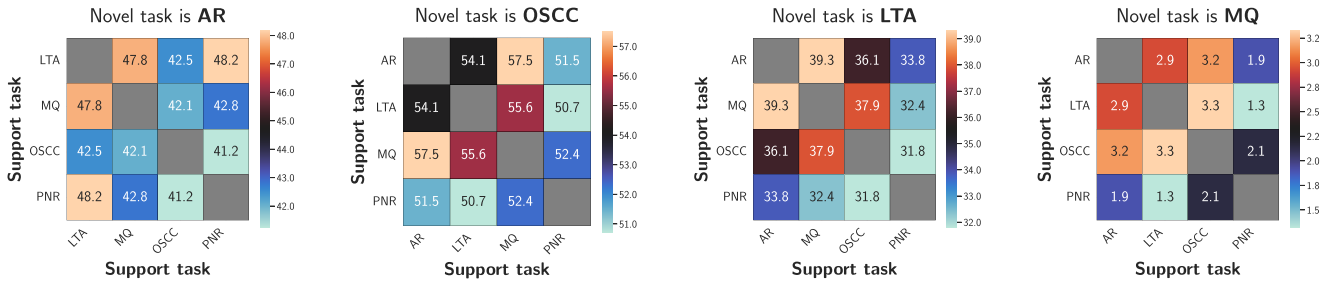

Fig. 5. Activations consensus for different ***novel tasks*** **.** Activations consensus between two *support tasks* is defined as the percentage of their prototypes corresponding to the same label activated by the two tasks. Fine-grained tasks, i.e., AR, OSCC and LTA, have higher average consensus. On the contrary, MQ has lower average consensus.

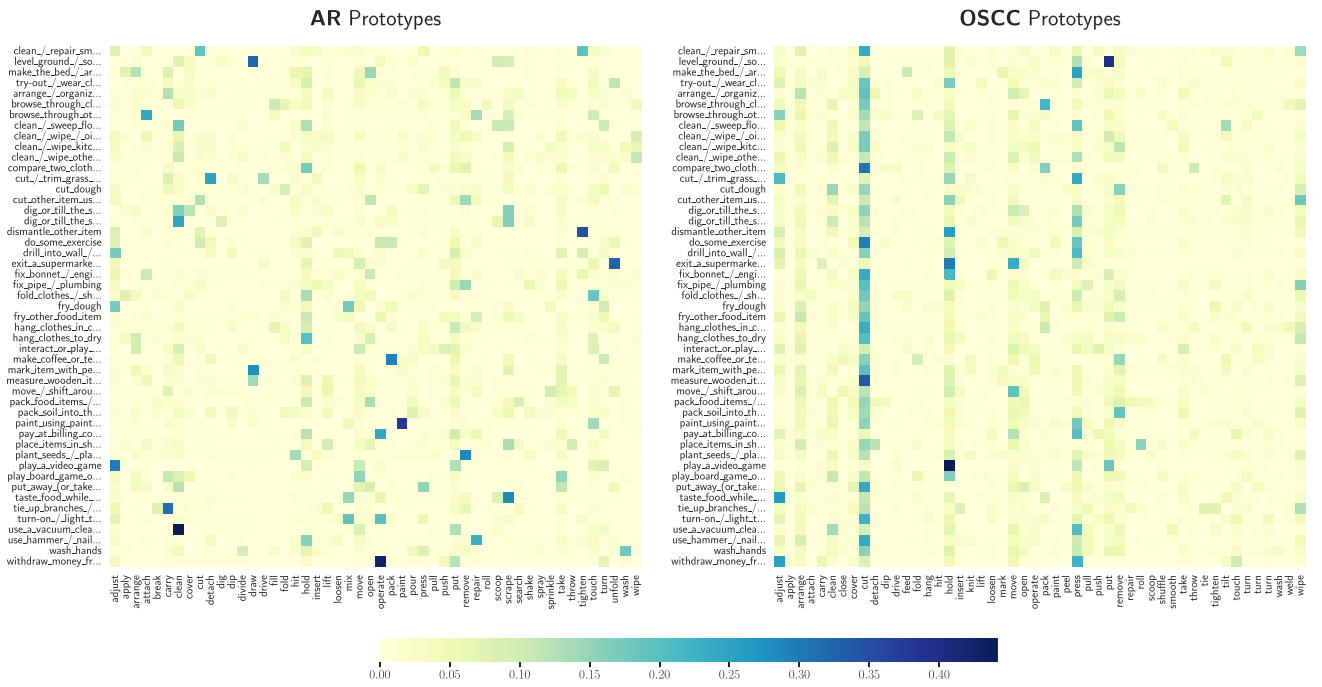

Fig. 6. Activation frequency of the prototypes from the ***support tasks*** **when the** ***novel task*** **is Moment Queries (MQ).** For each task from the MTL pre-training phase, we plot the distribution of closest prototypes in the Hier-EgoPack interaction phase. For readability, we restrict our analysis to the top 50 most predicted labels and activated prototypes. *Best viewed on a screen.*

difference to the implementation of the interaction process for these two groups of tasks. In fine-grained tasks, the interaction process is applied on the sample-level aligned features. On the contrary, for MQ the interaction is applied to node-level features, without any alignment due to the nature of the task, as previously stated in Section IV-B1. Therefore, a substantially higher number of nodes per video interact with the task-specific prototypes. These nodes may correspond to background regions of the video or to segments of an activity that are insufficiently discriminating. The low average activations consensus (Fig. 5)

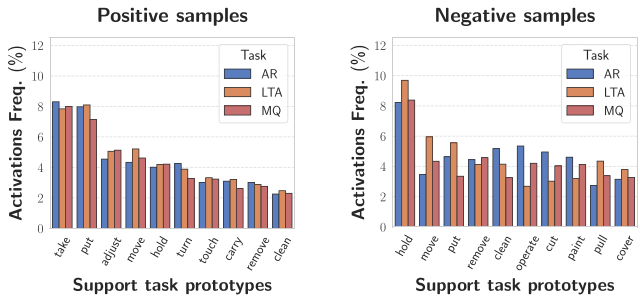

Fig. 7. Activation frequency of the prototypes from the ***support tasks*** **when the *novel task* is OSCC**, separately for the positive and negative correct predictions. Positive samples tend to focus more on prototypes whose verb could be associated with an object state change, e.g., *take* or *put*, compared to negative samples.

and high diversity in prototypes' activations across tasks (Fig. 4) show how Hier-EgoPack is effectively integrating different perspectives for the Moment Queries task.

Activation frequency: We show in Fig. 6 the most activated prototypes for different *support tasks* when the *novel task* is MQ. To enhance readability, we select the 50 most predicted labels and 50 most activated prototypes. Overall, we observe that the activations of the AR task are quite sparse, indicating that the novel task looks at very different perspectives from these tasks. On the contrary, the activations of the OSCC task are more uniform across different MQ labels. This is because these tasks focus on detecting object state changes in the video, which are typically associated with a subset of specific actions, such as *cut* or *mix*. As a result, only a subset of prototypes from these *support tasks* is actually activated by the novel task, as can be seen from the stripes in the plots.

Similarly, we show in Fig. 7 the most activated prototypes when the *novel task* is OSCC. We consider separately correctly predicted segments that contain an object state change (*positive*) or not (*negative*). Positive samples tend to focus more on prototypes whose verb could be associated with an object state change, e.g., *take* or *put*, compared to negative samples.

V. Limitations and Future Works

Hier-EgoPack requires labeled data from the novel task to align the prototypes' space with each task and to incorporate task-specific reasoning paradigms, e.g., temporal regression in MQ and PNR, preventing its extension to zero-shot scenarios. We argue that this limitation stems from the way these tasks are defined, rather than being an inherent limitation of our approach. Reformulating the tasks in a shared output space using language would eliminate the need for an alignment phase when dealing with a novel task. Alternatively, we could feed an LLM with the prototypes activations from a set of support tasks and a brief textual description of the novel task, leveraging the reasoning capabilities of the language model to associate the activations with the expected output for the task. During training, we could finetune the LLM on a set of *known novel tasks*, leveraging the textual task description to generalize to *unseen novel tasks* at inference time.

VI. Conclusion

We presented Hier-EgoPack, an extension of EgoPack that enables knowledge sharing between egocentric vision tasks with different temporal granularity. Hier-EgoPack is built on a unified temporal architecture that progressively learns more abstract representations of the input video, using a novel GNN layer specifically designed to incorporate strong temporal reasoning. We evaluate our approach in a *novel task learning* setting, in which a model is first trained on set of known *support tasks* and then has to leverage the knowledge obtained from such tasks to improve the learning process of a *novel task*. We validate Hier-EgoPack on five Ego4D tasks, covering a wide range of temporal granularities, from sub-second actions to long-range activities. Results show the effectiveness of our approach in knowledge reuse, outperforming single-task and multiple-task baselines, as well as task translation approaches that seek to share knowledge across tasks but lack explicit knowledge abstraction. Our work emphasizes the importance of prior knowledge and task perspectives in learning novel tasks, focusing on how task-specific knowledge is represented and utilized. Furthermore, through our proposed unified video understanding architecture, we demonstrate that leveraging diverse task perspectives in egocentric vision, even across varying temporal granularities, leads to more comprehensive and human-like video understanding.

Acknowledgment

The authors acknowledge the CINECA award under the ISCRA initiative, for the availability of high performance computing resources and support. This manuscript reflects only the authors. views and opinions, neither the European Union nor the European Commission can be considered responsible for them.

References

[1] S. Yan et al., "Multiview transformers for video recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 3323–3333.
[2] Z. Zhong, D. Schneider, M. Voit, R. Stiefelhagen, and J. Beyerer, "Anticipative feature fusion transformer for multi-modal action anticipation," in *Proc. IEEE/CVF Winter Conf. Appl. Comput. Vis.*, 2023, pp. 6057–6066.
[3] C.-L. Zhang, J. Wu, and Y. Li, "ActionFormer: Localizing moments of actions with transformers," in *Proc. Eur. Conf. Comput. Vis.*, 2022, pp. 492–510.
[4] I. Kokkinos, "UberNet: Training a universal convolutional neural network for low-, mid-, and high-level vision using diverse datasets and limited memory," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 5454–5463.
[5] Z. Xue, Y. Song, K. Grauman, and L. Torresani, "Egocentric video task translation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 2310–2320.
[6] S. A. Peirone, F. Pistilli, A. Alliegro, and G. Averta, "A backpack full of skills: Egocentric video understanding with diverse task perspectives," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2024, pp. 18275–18285.
[7] P. Bagad, M. Tapaswi, and C. G. Snoek, "Test of time: Instilling video-language models with a sense of time," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 2503–2516.

[8] K. Grauman et al., "Ego4D: Around the world in 3,000 hours of egocentric video," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 18973–18990.

[9] A. Betancourt, P. Morerio, C. S. Regazzoni, and M. Rauterberg, "The evolution of first person vision methods: A survey," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 25, no. 5, pp. 744–760, May 2015.

[10] C. Plizzari et al., "An outlook into the future of egocentric vision," *Int. J. Comput. Vis.*, vol. 132, pp. 4880–4936, 2024.

[11] D. Damen et al., "The EPIC-KITCHENS dataset: Collection, challenges and baselines," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 11, pp. 4125–4141, Nov. 2021.

[12] D. W. Hansen and Q. Ji, "In the eye of the beholder: A survey of models for eyes and gaze," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 3, pp. 478–500, Mar. 2010.

[13] Y. Jang, B. Sullivan, C. Ludwig, I. Gilchrist, D. Damen, and W. Mayol-Cuevas, "EPIC-tent: An egocentric video dataset for camping tent assembly," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. Workshop*, 2019, pp. 4461–4469.

[14] D. Damen et al., "Rescaling egocentric vision: Collection, pipeline and challenges for EPIC-KITCHENS-100," *Int. J. Comput. Vis.*, vol. 130, pp. 33–55, 2022.

[15] F. Sener et al., "Assembly101: A large-scale multi-view video dataset for understanding procedural activities," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 21064–21074.

[16] A. Núñez-Marcos, G. Azkune, and I. Arganda-Carreras, "Egocentric vision-based action recognition: A survey," *Neurocomputing*, vol. 472, pp. 175–197, 2022.

[17] A. Furnari and G. M. Farinella, "Rolling-unrolling LSTMs for action anticipation from first-person video," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 11, pp. 4021–4036, Nov. 2021.

[18] R. Girdhar and K. Grauman, "Anticipative video transformer," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 13485–13495.

[19] A. Furnari, S. Battiato, K. Grauman, and G. M. Farinella, "Next-active-object prediction from egocentric videos," *J. Vis. Commun. Image Representation*, vol. 49, pp. 401–411, 2017.

[20] Y. Huang, Y. Sugano, and Y. Sato, "Improving action segmentation via graph-based temporal reasoning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 14021–14031.

[21] S. K. Ramakrishnan, Z. Al-Halah, and K. Grauman, "SpotEM: Efficient video search for episodic memory," in *Proc. Int. Conf. Learn. Representations*, 2023, pp. 28618–28636.

[22] G. Goletto, T. Nagarajan, G. Averta, and D. Damen, "AMEGO: Active memory from long egocentric videos," in *Proc. Eur. Conf. Comput. Vis.*, 2024, pp. 92–110.

[23] K. Mangalam, R. Akshulakov, and J. Malik, "EgoSchema: A diagnostic benchmark for very long-form video language understanding," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2023, pp. 46212–46244.

[24] B. Jia, T. Lei, S.-C. Zhu, and S. Huang, "EgoTaskQA: Understanding human tasks in egocentric videos," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2022, pp. 3343–3360.

[25] J. Munro and D. Damen, "Multi-modal domain adaptation for fine-grained action recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 119–129.

[26] L. Yang, Y. Huang, Y. Sugano, and Y. Sato, "Interact before align: Leveraging cross-modal knowledge for domain adaptive action recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 14702–14712.

[27] M.-H. Chen, Z. Kira, G. AlRegib, J. Yoo, R. Chen, and J. Zheng, "Temporal attentive alignment for large-scale video domain adaptation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 6320–6329.

[28] C. Plizzari, T. Perrett, B. Caputo, and D. Damen, "What can a cook in Italy teach a mechanic in India? Action recognition generalisation over scenarios and locations," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2023, pp. 13610–13620.

[29] M. Planamente, C. Plizzari, S. A. Peirone, B. Caputo, and A. Bottino, "Relative norm alignment for tackling domain shift in deep multi-modal classification," *Int. J. Comput. Vis.*, vol. 132, pp. 2618–2638, 2024.

[30] Z. Sun, Q. Ke, H. Rahmani, M. Bennamoun, G. Wang, and J. Liu, "Human action recognition from various data modalities: A review," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 3, pp. 3200–3225, Mar. 2023.

[31] R. Gao, T.-H. Oh, K. Grauman, and L. Torresani, "Listen to look: Action recognition by previewing audio," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 10454–10464.

[32] K. Q. Lin et al., "Egocentric video-language pretraining," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2022, pp. 7575–7586.

[33] S. Pramanick et al., "EgoVLPv2: Egocentric video-language pre-training with fusion in the backbone," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2023, pp. 5262–5274.

[34] K. Ashutosh, R. Girdhar, L. Torresani, and K. Grauman, "HierVL: Learning hierarchical video-language embeddings," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 23066–23078.

[35] Y. Zhao, I. Misra, P. Krähenbühl, and R. Girdhar, "Learning video representations from large language models," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 6586–6597.

[36] Y. Tang et al., "Video understanding with large language models: A survey," *IEEE Trans. Circuits Syst. Video Technol.*, early access, May 02, 2025, doi: 10.1109/TCSVT.2025.3566695.

[37] M. Maaz, H. Rasheed, S. Khan, and F. Khan, "Video-ChatGPT: Towards detailed video understanding via large vision and language models," in *Proc. 62nd Annu. Meeting Assoc. Comput. Linguistics*, 2024, pp. 12585–12602.

[38] S. Ren, L. Yao, S. Li, X. Sun, and L. Hou, "TimeChat: A time-sensitive multimodal large language model for long video understanding," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2024, pp. 14313–14323.

[39] Q. Zhao et al., "AntGPT: Can large language models help long-term action anticipation from videos?," in *Proc. Int. Conf. Learn. Representations*, 2024.

[40] Y. Lu, Y. Song, W. Wang, L. Torresani, and T. Nagarajan, "VITED: Video temporal evidence distillation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2025, pp. 8501–8511.

[41] Y. Liu et al., "TempCompass: Do video LLMs really understand videos?," in *Proc. Findings Assoc. Comput. Linguistics*, 2024, pp. 8731–8772.

[42] C. Plizzari, A. Tonioni, Y. Xian, A. Kulshrestha, and F. Tombari, "Omnia de EgoTempo: Benchmarking temporal understanding of multi-modal LLMs in egocentric videos," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2025, pp. 24129–24138.

[43] H. Liu, C. Li, Q. Wu, and Y. J. Lee, "Visual instruction tuning," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2023, pp. 34892–34916.

[44] B. Li et al., "LLaVA-OneVision: Easy visual task transfer," *Trans. Mach. Learn. Res.*, 2025.

[45] B. Lin et al., "Video-LLaVA: Learning united visual representation by alignment before projection," in *Proc. Conf. Empir. Methods Natural Lang. Process.*, 2024, pp. 5971–5984.

[46] H. Zhang, X. Li, and L. Bing, "Video-LLaMA: An instruction-tuned audio-visual language model for video understanding," in *Proc. 2023 Conf. Empir. Methods Natural Lang. Process.: Syst. Demonstrations*, 2023, pp. 543–553.

[47] A. Hurst et al., "GPT-4o system card," 2024, *arXiv:2410.21276*.

[48] S. Bai et al., "Qwen2.5-VL technical report," 2025, *arXiv:2502.13923*.

[49] Z. Li, F. Liu, W. Yang, S. Peng, and J. Zhou, "A survey of convolutional neural networks: Analysis, applications, and prospects," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 12, pp. 6999–7019, Dec. 2022.

[50] A. Khan, A. Sohail, U. Zahoora, and A. S. Qureshi, "A survey of the recent architectures of deep convolutional neural networks," *Artif. Intell. Rev.*, vol. 53, pp. 5455–5516, 2020.

[51] J. Gu et al., "Recent advances in convolutional neural networks," *Pattern Recognit.*, vol. 77, pp. 354–377, 2018.

[52] M. Simonovsky and N. Komodakis, "Dynamic edge-conditioned filters in convolutional neural networks on graphs," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 29–38.

[53] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic graph CNN for learning on point clouds," *ACM Trans. Graph.*, vol. 38, 2019, Art. no. 146.

[54] F. Pistilli and G. Averta, "Graph learning in robotics: A survey," *IEEE Access*, vol. 11, pp. 112664–112681, 2023.

[55] S. Kearnes, K. McCloskey, M. Berndl, V. Pande, and P. Riley, "Molecular graph convolutions: Moving beyond fingerprints," *J. Comput.-Aided Mol. Des.*, vol. 30, pp. 595–608, 2016.

[56] W. Fan et al., "Graph neural networks for social recommendation," in *Proc. World Wide Web Conf.*, 2019, pp. 417–426.

[57] A. Sanchez-Gonzalez, J. Godwin, T. Pfaff, R. Ying, J. Leskovec, and P. Battaglia, "Learning to simulate complex physics with graph networks," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 8459–8468.

[58] R. Zeng et al., "Graph convolutional networks for temporal action localization," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 7093–7102.

[59] P. Ghosh, Y. Yao, L. Davis, and A. Divakaran, "Stacked spatio-temporal graph convolutional networks for action segmentation," in *Proc. IEEE/CVF Winter Conf. Appl. Comput. Vis.*, 2020, pp. 565–574.
[60] M. Rashid, H. Kjellstrom, and Y. J. Lee, "Action Graphs: Weakly-supervised action localization with graph convolution networks," in *Proc. IEEE/CVF Winter Conf. Appl. Comput. Vis.*, 2020, pp. 604–613.
[61] P. Ghosh, N. Saini, L. S. Davis, and A. Shrivastava, "All about knowledge graphs for actions," 2020, *arXiv: 2008.12432*.
[62] E. Dessalene, M. Maynord, C. Devaraj, C. Fermuller, and Y. Aloimonos, "Egocentric object manipulation graphs," 2020, *arXiv: 2006.03201*.
[63] E. Dessalene, C. Devaraj, M. Maynord, C. Fermuller, and Y. Aloimonos, "Forecasting action through contact representations from first person video," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 6, pp. 6703–6714, Jun. 2023.
[64] T. Nagarajan, Y. Li, C. Feichtenhofer, and K. Grauman, "EGO-TOPO: Environment affordances from egocentric video," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 160–169.
[65] R. Caruana, "Multitask learning," *Mach. Learn.*, vol. 28, pp. 41–75, 1997.
[66] Y. Zhang and Q. Yang, "A survey on multi-task learning," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 12, pp. 5586–5609, Dec. 2022.
[67] T. E. Huang, Y. Liu, L. Van Gool, and F. Yu, "Video task decathlon: Unifying image and video tasks in autonomous driving," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2023, pp. 8613–8623.
[68] Y. Huang, M. Cai, Z. Li, F. Lu, and Y. Sato, "Mutual context network for jointly estimating egocentric gaze and action," *IEEE Trans. Image Process.*, vol. 29, pp. 7795–7806, 2020.
[69] C. Fifty, E. Amid, Z. Zhao, T. Yu, R. Anil, and C. Finn, "Efficiently identifying task groupings for multi-task learning," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2021, pp. 27503–27516.
[70] T. Chen, S. Saxena, L. Li, T.-Y. Lin, D. J. Fleet, and G. E. Hinton, "A unified sequence interface for vision tasks," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2022, pp. 31333–31346.
[71] T. Chen et al., "AdaMV-MoE: Adaptive multi-task vision mixture-of-experts," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2023, pp. 17300–17311.
[72] H. Shi, S. Ren, T. Zhang, and S. J. Pan, "Deep multitask learning with progressive parameter sharing," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2023, pp. 19867–19878.
[73] Y. Ci et al., "UniHCP: A unified model for human-centric perceptions," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 17840–17852.
[74] Z. Kang, K. Grauman, and F. Sha, "Learning with whom to share in multi-task feature learning," in *Proc. Int. Conf. Mach. Learn.*, 2011, pp. 521–528.
[75] P. Guo, C.-Y. Lee, and D. Ulbricht, "Learning to branch for multi-task learning," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 3854–3863.
[76] T. Standley, A. Zamir, D. Chen, L. Guibas, J. Malik, and S. Savarese, "Which tasks should be learned together in multi-task learning?," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 9120–9132.
[77] X. Sun, R. Panda, R. Feris, and K. Saenko, "AdaShare: Learning what to share for efficient deep multi-task learning," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2020, pp. 8728–8740.
[78] G. Kapidis, R. Poppe, E. van Dam, L. Noldus, and R. Veltkamp, "Multitask learning to improve egocentric action recognition," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. Workshop*, 2019, pp. 4396–4405.
[79] X. Wang, L. Zhu, H. Wang, and Y. Yang, "Interactive prototype learning for egocentric action recognition," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 8148–8157.
[80] A. Kendall, Y. Gal, and R. Cipolla, "Multi-task learning using uncertainty to weigh losses for scene geometry and semantics," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 7482–7491.
[81] Z. Chen, V. Badrinarayanan, C.-Y. Lee, and A. Rabinovich, "GradNorm: Gradient normalization for adaptive loss balancing in deep multitask networks," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 794–803.
[82] A. Sinha, Z. Chen, V. Badrinarayanan, and A. Rabinovich, "Gradient adversarial training of neural networks," 2018, *arXiv: 1806.08028*.
[83] M. Guo, A. Haque, D.-A. Huang, S. Yeung, and L. Fei-Fei, "Dynamic task prioritization for multitask learning," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 282–299.
[84] T. Yu, S. Kumar, A. Gupta, S. Levine, K. Hausman, and C. Finn, "Gradient surgery for multi-task learning," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2020, pp. 5824–5836.
[85] S. Vandenhende, S. Georgoulis, and L. Van Gool, "MTI-Net: Multi-scale task interaction networks for multi-task learning," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 527–543.
[86] S. Ruder, "An overview of multi-task learning in deep neural networks," 2017, *arXiv: 1706.05098*.
[87] C. Zhao, A. K. Thabet, and B. Ghanem, "Video self-stitching graph network for temporal action localization," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 13638–13647.
[88] R. Girdhar, M. Singh, N. Ravi, L. van der Maaten, A. Joulin, and I. Misra, "Omnivore: A single model for many visual modalities," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 16081–16091.
[89] J. Carreira and A. Zisserman, "Quo Vadis, action recognition? A new model and the Kinetics dataset," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 4724–4733.
[90] L. Sui, F. Mu, and Y. Li, "NMS threshold matters for ego4D moment queries–2nd place solution to the ego4D moment queries challenge 2023," 2023, *arXiv:2307.02025*.
[91] T.-Y. Ross and G. Dollár, "Focal loss for dense object detection," in *Proc. Int. Conf. Comput. Vis.*, 2017, pp. 2999–3007.
[92] Z. Zheng, P. Wang, W. Liu, J. Li, R. Ye, and D. Ren, "Distance-IoU loss: Faster and better learning for bounding box regression," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 12993–13000.
[93] C. Feichtenhofer, H. Fan, J. Malik, and K. He, "SlowFast networks for video recognition," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 6201–6210.
[94] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. Int. Conf. Learn. Representations*, 2017, pp. 11313–11320.
[95] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," in *Proc. Int. Conf. Learn. Representations*, 2018.
[96] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 1025–1035.
[97] T. Derr, Y. Ma, and J. Tang, "Signed graph convolutional networks," in *Proc. 2018 IEEE Int. Conf. Data Mining*, 2018, pp. 929–934.
[98] J. Shao, X. Wang, R. Quan, J. Zheng, J. Yang, and Y. Yang, "Action sensitivity learning for temporal action localization," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2023, pp. 13411–13423.
[99] J. Shao, X. Wang, R. Quan, and Y. Yang, "Action sensitivity learning for the ego4D episodic memory challenge 2023," 2023, *arXiv:2306.09172*.
[100] E. V. Mascaró, H. Ahn, and D. Lee, "Intention-conditioned long-term human egocentric action anticipation," in *Proc. IEEE/CVF Winter Conf. Appl. Comput. Vis.*, 2023, pp. 6037–6046.
[101] S. Kim, D. Huang, Y. Xian, O. Hilliges, L. Van Gool, and X. Wang, "PALM: Predicting actions through language models," in *Proc. Eur. Conf. Comput. Vis.*, 2024, pp. 140–158.

Simone Alberto Peirone received the BSc and MSc degrees in computer engineering from the Polytechnic University of Turin, in 2020 and 2022, respectively. He is currently working toward the PhD degree with the Visual and Multimodal Applied Learning Laboratory, Turin, working under the supervision of Prof. Giuseppe Averta. His research focuses on egocentric vision and applications of graph neural networks to video understanding.

Francesca Pistilli received the MSc degree in electronic engineering from the Polytechnic of Turin, in 2019, the MSc degree in electrical and computer engineering from the University of Illinois at Chicago, Chicago, Illinois, in 2020, and the PhD degree from the Image Processing and Learning Group (IPL), Polytechnic of Turin, in 2023. She is currently assistant professor with the Polytechnic of Turin. Her current research interests lie at the intersection between computer vision and robotics.

Antonio Alliegro is currently a postdoctoral researcher with the Polytechnic University of Turin. His research focuses on 3D understanding and its application to real-world scenarios, including research on reducing the synth-to-real domain gap and open-set 3D recognition. He has published multiple papers presented at prestigious computer vision conferences and journals such as CVPR, IROS, NeurIPS, and *IEEE Robotics and Automation Letters*. Additionally, he has contributed as a reviewer at various academic events.

Tatiana Tommasi received the PhD degree from EPFL Lausanne, in 2013. Subsequently, she undertook postdoctoral roles in both Belgium and the USA. She holds the position of associate professor with the Control and Computer Engineering Department, Polytechnic University of Turin and is director of the ELLIS Unit, Turin. Before her current position, she was an assistant professor with Sapienza University in Rome, Italy. Her publication record boasts more than 50 papers in top conferences and journals, specializing in machine learning and computer vision. Her expertise lies in the development of theoretically grounded algorithms for automatic learning from images, particularly within the realms of robotics, medical applications, and human–machine interaction. She pioneered the field of transfer learning in computer vision and possesses extensive experience in areas such as domain adaptation, generalization, multimodal learning, and open-set learning. She is an associate editor of *IEEE Transactions on Pattern Analysis and Machine Intelligence* and *IEEE Transactions on Emerging Topics in Computing*.

Giuseppe Averta received the PhD degree in robotics from the University of Pisa, in 2020. In 2019, he was a visiting student with the Eric P. and Evelyn E. Newman Laboratory, Biomechanics and Human Rehabilitation Group, MIT. He is currently an assistant professor of robotics and machine learning with the Polytechnic of Turin. He is also an Italian Institute of Technology Alumnus. His current research interests include the development of a truly embodied intelligence for human-robot cooperation, with research activities in human action recognition, deep learning for egocentric vision, human-inspired design, planning, and control guidelines for autonomous, collaborative, assistive, and prosthetic robots.