OBJECT-REGION VIDEO TRANSFORMERS

Anonymous authors
Paper under double-blind review

ABSTRACT

Evidence from cognitive psychology suggests that understanding spatio-temporal object interactions and dynamics can be essential for recognizing actions in complex videos. Therefore, action recognition models are expected to benefit from explicit modeling of objects, including their appearance, interaction, and dynamics. Recently, video transformers have shown great success in video understanding, exceeding CNN performance. Yet, existing video transformer models do not explicitly model objects. In this work, we present Object-Region Video Transformers (ORViT), an *object-centric* approach that extends video transformer layers with a block that directly incorporates object representations. The key idea is to fuse object-centric spatio-temporal representations throughout multiple transformer layers. Our ORViT block consists of two object-level streams: appearance and dynamics. In the appearance stream, an "Object-Region Attention" element applies self-attention over the patches and *object regions*. In this way, visual object regions interact with uniform patch tokens and enrich them with contextualized object information. We further model object dynamics via a separate "Object-Dynamics Module", which captures trajectory interactions, and show how to integrate the two streams. We evaluate our model on standard and compositional action recognition on Something-Something V2, standard action recognition on Epic-Kitchen100 and Diving48, and spatio-temporal action detection on AVA. We show strong improvement in performance across all tasks and datasets considered, demonstrating the value of a model that incorporates object representations into a transformer architecture.

1 INTRODUCTION

Consider the simple action of "Picking up a coffee cup" in Figure 1. Intuitively, a human recognizing this action would identify the hand, the coffee cup and the coaster, and perceive the upward movement of the cup. This highlights three important cues needed for recognizing actions: what/where are the objects? how do they interact? and how do they move? Indeed, evidence from cognitive psychology also supports this structure of the action-perception system (Quinton, 1979; Hacker, 1982; Gallese et al., 1996; Chao & Martin, 2000; Helbig et al., 2006). The above perception process allows easy generalization to different compositions of actions. For example, the process of picking up a knife will share some of the components with "Picking up a coffee cup". More broadly, representing image semantics using objects facilitates compositional understanding, because many perceptual components remain similar when one object is swapped for another. Thus, a model that captures this compositional aspect potentially requires less examples to learn.

It seems intuitively clear that machine vision models should also capture the above reasoning structure, and indeed this has been explored in the past (Gupta & Davis, 2007; Saenko et al., 2012). However, current state-of-the-art video transformer models do not explicitly model objects. Recently, video transformers have been introduced as a powerful video understanding models (Arnab et al., 2021b; Bertasius et al., 2021; Haoqi et al., 2021; Patrick et al., 2021), motivated by the success of transformers in language (Devlin et al., 2019) and vision (Carion et al., 2020; Dosovitskiy et al., 2021). In these models, each video frame is divided into patches, and a spatio-temporal self-attention architecture obtains a context-dependent representation for the patches. However, there is no explicit representation of objects in this approach, which makes it harder for such models to capture compositionality.

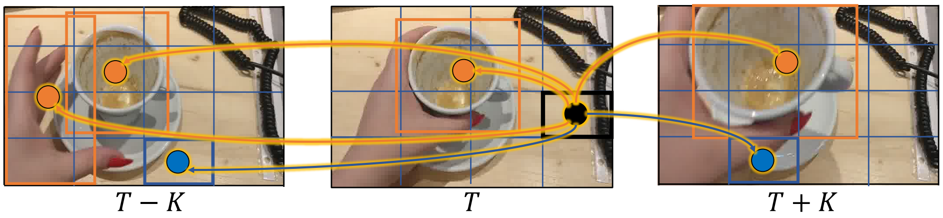

Figure 1: Our ORViT model introduces *class-agnostic* object-centric information into the transformer self-attention operation. The figure shows the standard (uniformly spaced) patch-tokens in **blue**, and object-regions corresponding to class-agnostic detections in **orange**. In ORViT any temporal patch-token (e.g., the patch in **black** at time T) attends to all patch tokens (**blue**) and region tokens (**orange**). This allows the new patch representation to be informed by the objects.

Motivated by the above, our key question in this paper is how to model objects explicitly within a video-transformer (Arnab et al., 2021b) architecture. We propose a general approach by adapting the self-attention block (Dosovitskiy et al., 2021) to incorporate object information. The challenge in building such an architecture is that it should have components for modeling the appearance of objects as they move, the interaction between objects, and the dynamics of the objects (irrespective of their visual appearance). An additional key desideratum is that video content outside the objects should not be discarded, as it contains important contextual information. Finally, we would like the objects to influence the representation of the scene, throughout the bottom-up process, rather than as a post-processing stage. In what follows, we show that a self-attention architecture can be extended to address these aspects. Our key idea is that object regions can be introduced into transformers in a similar way to that of the regular patches, and dynamics can also be integrated into this framework in a natural way. We refer to our model as an "Object-Region Video Transformer" (or ORViT).

The ORViT block takes as input patch tokens (also referred as spatio-temporal features) and outputs refined patch tokens based on object information. Within the block, it uses a set of object bounding boxes that are predicted using largely *class-agnostic* off-the-shelf trackers, and serve to inform the model which parts of video contain objects. This information is then used to generate two separate object-level streams: an "Object-Region Attention" stream that models appearance, and an "Object-Dynamics Module" stream that models trajectories.[1] The appearance stream first extracts descriptors for each object based on the object coordinates and the patch tokens. Next, we append the object descriptors to the patch tokens, and a self-attention is applied to all these tokens jointly, and thus incorporating object information into the patch tokens (see Figure 1). The trajectory stream only uses object coordinates to model the geometry of motion and performs self-attention over those. Finally, we re-integrate the appearance and trajectory stream into a set of refined patch tokens, which have the same dimensionality as the input to our ORViT block. This means that the ORViT block can be called repeatedly. See Figure 2 and Figure 3 for the model visualizations.

Through extensive empirical study, we show that integrating the ORViT block into video transformer architecture leads to improved Action Recognition results on Something-Something, Epic-Kitchens100, Diving48, improved Spatio-temporal Action Detection on AVA as well as improved Compositional Action Recognition on SomethingElse. These results confirm our hypothesis that explicitly modeling objects indeed leads to better generalization.

2 Related Work

Object-centric models. Recently object-centric models have been successfully applied in many computer vision applications: visual relational reasoning (Battaglia et al., 2018; Zambaldi et al., 2018; Krishna et al., 2018; Baradel et al., 2018; Herzig et al., 2018; Xu et al., 2020; Raboh et al., 2020; Jerbi et al., 2020), representation learning (You et al., 2020), vision and language (Li et al., 2019; Tan & Bansal, 2019; Li et al., 2020; Chen et al., 2020), human-object interactions (Kato et al., 2018; Xu et al., 2019; Gao et al., 2020), and even image generation (Johnson et al., 2018; Herzig et al., 2020). The advances and the success of object-centric models in these domains inspired varied video-based tasks, such as action localization (Nawhal & Mori, 2021), video synthesis (Bar et al., 2021), and action recognition (Zhou et al., 2018). The latter was the focus of varied recent

[1] Our focus is different from papers on two-stream models in vision, that are not object-centric (see Sec. 2).

works that designed different object interactions approaches for convolutional models. A line of works (Santoro et al., 2017; Sun et al., 2018; Girdhar et al., 2019) focused on capturing spatial object interactions while ignoring the temporal interactions. STRG (Wang & Gupta, 2018) and ORN (Baradel et al., 2018) used spatio-temporal interactions with two consecutive frame interactions, while STAG (Herzig et al., 2019) considered long-range temporal interactions. Last, Unified (Arnab et al., 2021a) tried to generalize all these models and propose long spatio-temporal object interactions. While all these works focused solely on interactions of visual appearance information, recently STIN (Materzynska et al., 2020) introduced an object-centric model based on object trajectories by modeling bounding box movement. Our ORViT approach directly combines object appearance, object trajectories, and the overall video, by mapping all computations to spatio-temporal patch tokens. This is particularly natural to do in a transformer framework, as we show here, and results in state of the art performance. We note that models combining vision and language (e.g., Li et al., 2020) use detectors to extract object features and utilize them later in their network. Our approach is focused on video and markedly different from theirs since we are not extracting detection features.

Transformers in action recognition. Ranging from the early works that employ optical flow based features (Efros et al., 2003), to recent transformer based approaches (Haoqi et al., 2021), a wide variety of approaches have been proposed for action recognition. In broad brushstrokes, the proposed approaches have evolved from using temporal pooling for extracting features (Karpathy et al., 2014) to using recurrent networks (Donahue et al., 2015; Yue-Hei Ng et al., 2015), through to 3D spatio-temporal kernels (Ji et al., 2013; Taylor et al., 2010; Tran et al., 2015; Varol et al., 2018; Lin et al., 2019a; Wang et al., 2019; Carreira & Zisserman, 2017), and two-stream networks that capture complementary signals (e.g., motion and spatial cues (Feichtenhofer et al., 2016; Simonyan & Zisserman, 2014; Feichtenhofer et al., 2019)). Unlike these approaches, our work uses two separate object-level streams to leverage object-centric information. In parallel to developments in video understanding, Vision Transformers (Dosovitskiy et al., 2021; Touvron et al., 2021) propose a new approach to image recognition by discarding the convolutional inductive bias entirely and instead employing self-attention operations. Specialized video models such as TimeSformer (Bertasius et al., 2021), ViViT (Arnab et al., 2021b), Mformer (Patrick et al., 2021) and MViT (Haoqi et al., 2021) form the latest epoch in action recognition models. However, none of the video transformer models leverage object cues, a persistent shortcoming that we aim to address in ORViT.

Spatio-temporal action detection. The task of action detection requires temporally localizing the action start and end times. A wide variety of methods have been proposed for it, such as actions modeling (Long et al., 2019; Zeng et al., 2019; Alwassel et al., 2018), temporal convolutions (Shou et al., 2016; Lea et al., 2017), boundaries modeling (Lin et al., 2018; 2019b), attention (Shi et al., 2020; Yuan et al., 2019), structure utilization (Yuan et al., 2017; Zhao et al., 2017), detection based methods (Chao et al., 2018; Xu et al., 2017), end-to-end approaches (Dai et al., 2017; Gao et al., 2017; Buch et al., 2017; Jain et al., 2020), recurrent neural networks (Ma et al., 2016; Singh et al., 2016; Yeung et al., 2016), and even using language (Richard & Gall, 2016; Zhukov et al., 2019). Recently, the new MViT (Haoqi et al., 2021) model showed promising results on action localization in the AVA dataset (Gu et al., 2018b). However, it does not explicitly model objects, and we show that an ORViT version of MViT indeed improves performance.

3 THE ORVIT MODEL

We now present Object-Region Video Transformer (ORViT) model, which explicitly models object appearance and trajectories within the transformer architecture. We begin by reviewing the video transformer architecture in Section 3.1, which our model extends, and present ORViT in Section 3.2. A high-level overview of ORViT is shown in Figure 2 (left) and detailed in Figure 3. Briefly, ORViT repeatedly refines the patch token representations by using information about both the appearance and movement of objects. Figure 2 (right) visualizes object-centric attention learned by our model.

3.1 VIDEO TRANSFORMER ARCHITECTURE

Video transformers (Arnab et al., 2021b; Haoqi et al., 2021; Bertasius et al., 2021) extend the Vision Transformer model to the temporal domain. Similar to vision transformers, the input is first "patchified" but with temporally extended 3-D patches instead of 2-D image patches producing a down-sampled tensor X of size, say, $T \times H \times W \times d$. Then spatio-temporal position embeddings

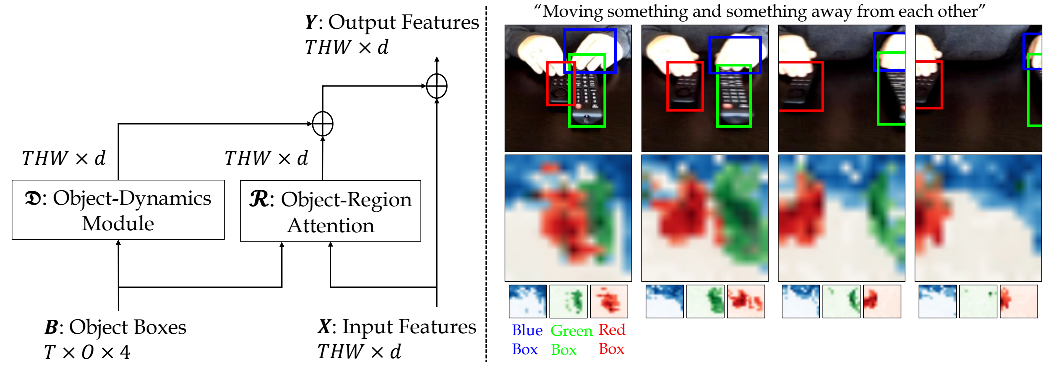

Figure 2: **Left**: An *ORViT block*. The input patch-tokens X and boxes B are used as input to the "Object-Region Attention" and "Object-Dynamics Module" components. Each component outputs a $THW \times d$ tensor and the two tensors are summed to obtain new patch tokens Y. **Right**: We visualize the attention allocated to the object tokens in the ORViT block (red, green, and blue) in each frame for a video describing "moving two objects away from each other". It can be seen that each phone object affects the patch-tokens in its region, whereas the hand has a broader map. For more visualizations, please see section E in supplementary.

are added in for location information. Finally, a classification token (CLS) is appended to X, resulting in $THW + 1$ tokens in $\mathbb{R}^d$, to which self-attention and MLP blocks are applied repeatedly to produce the final contextualized CLS feature vector which is used for classification.[2]

3.2 The ORViT Block

There are two inputs to the ORViT block. The first is the output of the preceding transformer block, represented as a set of spatio-temporal tokens $X \in \mathbb{R}^{THW \times d}$. The second input is a set of bounding boxes for objects across time, provided by an off-the-shelf object detector[3] or available directly in the dataset. These bounding boxes are denoted by $B \in \mathbb{R}^{TO \times 4}$ (providing the bounding box for each object at each time frame). The output of the ORViT block is a set of refined tokens $Y \in \mathbb{R}^{THW \times d}$ contextualized with object-centric information. Thus, the block can be viewed as a token representation refining mechanism using the object-level information.

As mentioned, we argue that the key cues for recognizing actions in videos are: the objects in the scene, their interactions, and their movement. To capture these cues, we design the ORViT block with the following two object-level streams. The first stream models the appearance of objects, and their interactions. We refer to it as "Object-Region Attention" and denote it by $\mathcal{R}$. The second "Object-Dynamics Module" stream (denoted by $\mathcal{D}$) models the interactions between trajectories, agnostic of their appearance. Importantly, the output of each of the streams is THW token vectors, which can also be interpreted as refined patch representations based on each source of information.

The $\mathcal{D}$ stream only models object dynamics, and thus only uses bounding boxes, B as input. We therefore denote its output by $\mathcal{D}(B)$. The stream $\mathcal{R}$ models appearance and thus depends on both the token represention, X and the bounding boxes, B and produces $\mathcal{R}(X, B)$. The final output of the ORViT block Y is simply formed by the sum of the two streams and an input residual connection:

$$Y' := \mathcal{R}(X, B) + \mathcal{D}(B) + X \quad , \quad Y := Y' + \text{MLP}(\text{LN}(Y')) \tag{1}$$

where LN denotes a LayerNorm operation. Next, we elaborate on the two components separately.

Object-Region Attention. The goal of this module is to extract information about each object and use it to refine the patch tokens. This is done by using the object regions to extract descriptor vectors per region from the input tokens, resulting in TO vectors in $\mathbb{R}^d$, which we refer to as object tokens. These vectors are then concatenated with the THW patch tokens and serve as the keys and values, while the queries are only the patch tokens. Finally, the output of the block is THW patch tokens. Thus, the key idea is to fuse object-centric information into spatio-temporal representations.

[2] In what follows we omit the count of the CLS feature for brevity.

[3] We also use a simple tracker on top of the detections, but discuss detections here for brevity.

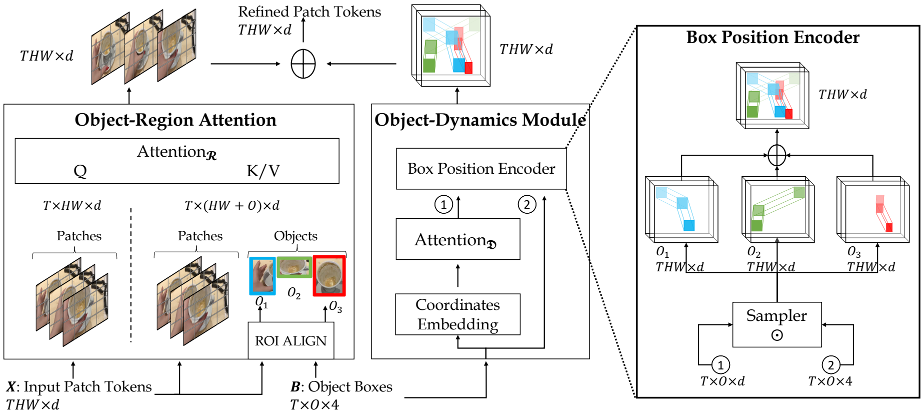

Figure 3: **ORViT Block architecture**. The block consists of two object-level streams: an "Object-Region Attention" that models appearance, and an "Object-Dynamics Module" that models trajectories. The two are combined to produce new patch tokens. The "Box Position Encoder" maps the output of the trajectory stream to the dimensional of patch tokens.

Namely, inject the TO object region tokens into patch tokens THW. An overview of our approach is depicted in Figure 3. We provide further details below.

Given the patch token features X and the boxes B, our first goal is to obtain vector descriptors in $\mathbb{R}^d$ per object and time frame. The natural way to do this is via an RoIAlign (He et al., 2017) layer, which uses the patch tokens X and box coordinates B to obtain object region crops. This is followed by max-pooling and an MLP to get the final object representation in $\mathbb{R}^d$:

$$\mathcal{O} := \text{MLP}(\text{MaxPool}(\text{RoIAlign}(X, B))) \tag{2}$$

Since this is done per object and per frame, the result is OT vectors in $\mathbb{R}^d$ (i.e., $\mathcal{O} \in \mathbb{R}^{TO\times d}$). Importantly, this extraction procedure is performed in *each instance of an ORViT block*, so that it will produce different object tokens at each layer. We also add positional embeddings but leave the details to Section B.1.

At this point, we would like to allow the object tokens to refine the patch tokens. We concatenate the object tokens $\mathcal{O}$ with the patch tokens X, resulting in $C \in \mathbb{R}^{T(HW+O)\times d}$. Next C and X are used to obtain queries, keys and values as follows:

$$Q := XW_q \quad K := CW_k \quad V := CW_v \quad \text{, where } W_q, W_k, W_v \in \mathbb{R}^{d\times d} \tag{3}$$

Finally, there are several ways to perform spatio-temporal self-attention (e.g., joint attention over space and time, attention over space and then time, or the recently introduced trajectory attention (Patrick et al., 2021)). We use trajectory attention because it performs well empirically. We compare different self-attention versions in Table 4c.

Object-Dynamics Module. To model object dynamics, we introduce a component that only considers the boxes B. We first encode each box via its center coordinate, height and width, and apply an MLP to this vector to obtain a vector in $\mathbb{R}^d$. Applying this to all boxes results in $\widetilde{L} \in \mathbb{R}^{TO\times d}$. Next we add a learnable object-time position embedding $\widetilde{P} \in \mathbb{R}^{TO\times d}$, resulting in $\widetilde{B} := \widetilde{L} + \widetilde{P}$. We refer to this as the "Coordinate Embedding" step in Figure 3. Its output can be viewed as TO tokens in $\mathbb{R}^d$, and we apply self-attention to those as follows: $\text{Attention}_{\mathcal{D}}(\widetilde{Q}, \widetilde{K}, \widetilde{V}) := \text{Softmax}\left(\frac{\widetilde{Q}\widetilde{K}^T}{\sqrt{d_k}}\right)\widetilde{V}$, where: $\widetilde{Q} := \widetilde{B}W_{\widetilde{q}}$, $\widetilde{K} := \widetilde{B}W_{\widetilde{k}}$, $\widetilde{V} := \widetilde{B}W_{\widetilde{v}}$ and $W_{\widetilde{q}}, W_{\widetilde{k}}, W_{\widetilde{v}} \in \mathbb{R}^{d\times d}$. The self-attention output is in $\mathbb{R}^{TO\times d}$. Next, we would like to transform the objects with a $T \times d$ vector into a spatial volume of $THW \times d$. This is done using the Box Position Encoder described below.

Box Position Encoder. The returned features of the ORViT model should have the same dimensions as the input, namely $THW \times d$. Thus, our main challenge is to project the object embeddings into spatial dimensions, namely $TO \times d$ into $THW \times d$. The naive approach would be to ignore the boxes by expanding every object with vector $T \times d$ into $THW \times d$. However, since the object trajectories contain their space-time location, a potentially better way to do it would consider the object locations. Hence, for each object with corresponding $T \times d$ tokens, we generate a spatial feature $HW \times d$ by placing the object representation vector according to the matching bounding box coordinates using a bilinear interpolation sampler operation[4] (Jaderberg et al., 2015; Johnson et al., 2018). Finally, the output in $HW \times d$ is the sum of all objects from all frames representing the coarse trajectory of the object in spatial dimensions. The process is shown in Figure 3 (right). We show empirically that this approach is better than the naive approach described above.

The ORViT model: We conclude by explaining how to integrate ORViT into transformer-based video models. The advantage of ORViT is that it takes as input the standard spatio temporal tokens in $\mathbb{R}^{THW \times d}$ and outputs a refined version of those with the same dimensions. Thus, it acts as a standard transformer layer in terms of input and output. Therefore, one can take any transformer and simply add ORViT layers to it. In particular, we experiment with three such models: TimeSformer (Bertasius et al., 2021), Mformer, and MViT (Haoqi et al., 2021). We show that for these, using ORViT layers improves performance. The only design choice is which layers to apply ORViT to. We found that it is very important to apply it in lower layers, while repeated applications further improves performance. Specifically, in our experiments we apply it in layers $2, 7, 11$.

4 EXPERIMENTS

We evaluate our ORViT block on several video action recognition benchmarks. Specifically, we consider the following tasks: Compositional Action Recognition (Section 4.1), Action Recognition (Section 4.2) and Spatio-Temporal Action Detection (Section 4.3).

Datasets: We experiment with the following datasets: **(1) Something-Something v2 (SSv2)** (Goyal et al., 2017) contains 174 action categories of common human-object interactions. We follow the official splits from (Materzynska et al., 2020) for action recognition, compositional action recognition, and few-shot recognition. **(2) Epic Kitchens 100 (EK100)** (Damen et al., 2020) contains 700 egocentric videos of daily kitchen activities. This dataset includes noun and verb classes, and we report verb, noun, and action accuracy, where the highest-scoring verb and noun pair constitutes an action label. **(3) Diving48** (Li et al., 2018) contains 48 fine-grained categories of diving activities. **(4) Atomic Visual Actions (AVA)** (Gu et al., 2018a) is a benchmark for human action detection. We report Mean Average Precision (mAP) on AVA-V2.2.

Baselines. In the experiments, we compare ORViT to several models reported in previous work for the corresponding datasets. These include non-transformer approaches (e.g., *I3D* (Carreira & Zisserman, 2017) and *SlowFast* (Feichtenhofer et al., 2019)) as well as state-of-the-art transformers (TimeSformer, Mformer, and MViT). We also cite results for two object-centric models: *STIN* (Materzynska et al., 2020) which uses boxes information, and the Space-Time Region Graph (STRG) model (Wang & Gupta, 2018) which extracts I3D features for objects and runs a graph neural network on those. Both STIN and STRG use the same input information as ORViT. Finally, we implement an object-centric transformer baseline combining STRG and STIN: we use the Mformer final patch tokens as input to the STRG model, resulting in STRG feature vector, and concatenate it to the STIN feature vector and the Mformer's CLS token. We refer to this as *Mformer+STRG+STIN*.

Box Input to ORViT: The ORViT block takes bounding boxes of objects as inputs. Here we explain how these are extracted in our experiments. We emphasize that in all cases, these are either bounding boxes provided with the dataset or the result of an off-the-shelf class-agnostic detector. For SSv2, we used both GT and detected boxes provided by the dataset. For EK100, we used detections provided by the dataset that were based on a pre-trained detector. For Diving48, we used a standard pre-trained class-agnostic detector from Detectron2 (trained on MSCOCO). For AVA, we use the provided detection boxes for the spatio-temporal action detection task. For all datasets, once we have the detector results, we apply multi-object tracking to find correspondence between the objects in different frames (no training data is required). See Section A.1.

Implementation Details: ORViT is implemented in PyTorch, and code will be released upon acceptance. Our training recipes and code are based on the MViT, Mformer, and TimeSformer code published by the authors. We set the number of objects to 4 in SSv2 and EK100, 6 in AVA, and 10 in Diving48. See Section A in Supplementary.

[4] Features outside of an object region are set to zeros.

Table 1: **Compositional and Few-Shot Action Recognition** on the "SomethingElse" dataset. All results are reported using ground-truth box annotations.

Model	Modality		Compositional		Base		Few-Shot	
	RGB	Boxes	Top-1	Top-5	Top-1	Top-5	5-Shot	10-Shot
I3D (Carreira & Zisserman, 2017)	✓	✗	42.8	71.3	73.6	92.2	21.8	26.7
SF (Feichtenhofer et al., 2019)	✓	✗	45.2	73.4	76.1	93.4	22.4	29.2
TimeSformer (Bertasius et al., 2021)	✓	✗	44.2	76.8	79.5	95.6	24.6	33.8
Mformer (Patrick et al., 2021)	✓	✗	60.2	85.8	82.8	96.2	28.9	33.8
STRG (\w SF) (Wang & Gupta, 2018)	✓	✓	52.3	78.3	75.4	92.7	24.8	29.9
STIN (\w SF) (Materzynska et al., 2020)	✓	✓	54.6	79.4	77.4	95.0	23.0	33.4
Mformer+STRG+STIN	✓	✓	62.3	86.0	83.7	96.8	29.8	36.5
ORViT Mformer (ours)	✓	✓	**69.7**	**91.0**	**87.1**	**97.6**	**33.3**	**40.2**

4.1 Compositional and Few-Shot Action Recognition

Several action recognition datasets define an action via a combination of a verb (action) and noun (object). In such cases, it is more challenging to recognize combinations that were not seen during training. This "compositional" setting was explored in the "SomethingElse" dataset (Materzynska et al., 2020), where verb-noun combinations in the test data do not occur in the training data. The split contains 174 classes with 54,919/54,876 videos for training/validation. This setting is of particular relevance for object-centric models like ORViT, which can potentially better handle compositional actions. Finally, the dataset also contains GT boxes, intended to specifically evaluate object-centric models that use this information. We also evaluate on the few-shot compositional action recognition task in (Materzynska et al., 2020) (see Section A.6 for details).

Table 1 reports the results in these settings, given GT boxes input. It can be seen that ORViT outperforms all models for both the *Compositional* and *Few-shot* tasks. Importantly, it provides large improvement over the baseline Mformer model. Recall that ORViT is essentially an Mformer with additional ORViT blocks, so the difference in performance is due to the ORViT block architecture and its use of box information. When using detected boxes (as opposed to GT), ORViT Mformer achieves 62.3% top-1 compared to the Mformer model with 60.2% top-1 on the compositional split.

4.2 Action Recognition

Table 2 reports results on the standard action recognition task for several datasets. We train on the standard splits and use the standard evaluation procedure. See Section A of Supplementary.

SSv2. Table 2a shows that ORViT outperforms the state-of-the-art with both detected and ground-truth (GT) boxes. GT boxes result in better performance, indicating the potential of using object-centric models like ORViT. When using GT boxes, improvements over *Mformer* and *Mformer-L* are 4.9% and 5.4%, respectively. When using detected boxes, improvement is 1.4% for both. ORViT also outperforms other box-based methods, such as *STIN*, *STRG* and their combination.

Diving48. Here we build ORViT on top of TimerSformer model, which was previously reported on this dataset (this demonstrates the ease of adding ORViT to any transformer model). Table 2b shows that our *ORViT TimeSformer* model outperforms the state-of-the-art methods, including TQN (Zhang et al., 2021) by a significant margin of 6.2%. We obtain an improvement of 8.0% over the baseline TimeSformer model to which ORViT blocks were added. This again indicates the direct improvement due to the ORViT block. We note that ORViT achieves these results using only 32 frames, significantly less than the previous best results, *TimeSformer-L*, which uses 96 frames.

EK100. Table 2c reports results on EK100. Here we add ORViT blocks to the *Mformer-HR* model. Results show that our *ORViT Mformer-HR* model improves the accuracy for all three tasks (with a smaller improvement for nouns). As in Diving48, we use only the detected boxes provided with the dataset (Damen et al., 2020), obtained with Faster-RCNN trained on MS COCO. We believe the improvements on EK100 are less impressive than on the other datasets for mainly two reasons: (a) EK100 is an ego-centric dataset, making the camera movement a significant challenge for our method to use objects effectively. Since the movement of the boxes is mainly due to the camera, it is more challenging to model meaningful object interactions as opposed to motion caused by objects movement. (b) EK100 contains short 2-3 seconds videos, thus temporal reasoning is less effective.

Table 2: **Comparison to state-of-the-art on video action recognition.** We report top-1 (%) and top-5 (%) accuracy on SSv2. On Epic-Kitchens100 (EK100), we report top-1 (%) action (A), verb (V), and noun (N) accuracy. On Diving48 we report top-1 (%). For EK100 and Diving48 we used only detected boxes since ground-truth does not exist. Difference between baselines and ORViT is denoted by (+X). IN refers to IN-21K.

(a) **Something–Something V2**

Model	Boxes	Pretrain	Top-1	Top-5	GFLOPs×views (10^9)	Param (10^6)
SlowFast, R101	✗	K400	63.1	87.6	106×3×1	53.3
TimeSformer-L	✗	IN	62.5	-	1703×3×1	121.4
ViViT-L	✗	IN+K400	65.4	89.8	3992×4×3	-
MViT-B, 64	✗	K600	68.7	91.5	236×3×1	53.2
Mformer	✗	IN+K400	66.5	90.1	369.5 × 3 × 1	109
Mformer-L	✗	IN+K400	68.1	91.2	1185.1 × 3 × 1	109
Mformer + STRG + STIN	GT	IN+K400	69.2	90.9	375 × 3 × 1	119
ORViT Mformer (Ours)	Detected	IN+K400	**67.9** (+1.4)	90.5 (+0.4)	405 × 3 × 1	148
ORViT Mformer (Ours)	GT	IN+K400	**73.8** (+7.3)	**93.6** (+3.5)	405 × 3 × 1	148
ORViT Mformer-L (Ours)	Detected	IN+K400	**69.5** (+1.4)	**91.5** (+0.3)	1259 × 3 × 1	148.2
ORViT Mformer-L (Ours)	GT	IN+K400	**74.9** (+6.7)	**94.2** (+3.0)	1259 × 3 × 1	148.2

(b) **Diving48**

Model	Pretrain	Frames	Top-1
SlowFast, R101	K400	16	77.6
TimeSformer	IN	16	74.9
TimeSformer-L	IN	96	81.0
TQN	K400	ALL	81.8
TimeSformer	IN	32	80.0
TimeSformer + STRG + STIN	IN	32	83.5
ORViT TimeSformer (Ours)	IN	32	**88.0** (+8.0)

(c) **Epic-Kitchens100**

Method	Pretrain	A	V	N
SlowFast, R50	K400	38.5	65.6	50.0
ViViT-L	IN+K400	44.0	66.4	56.8
Mformer	IN+K400	43.1	66.7	56.5
Mformer-L	IN+K400	44.1	67.1	57.6
Mformer-HR	IN+K400	44.5	67.0	58.5
MF-HR + STRG + STIN	IN+K400	44.1	66.9	57.8
ORViT Mformer-HR (Ours)	IN+K400	**45.7** (+1.2)	**68.4** (+1.4)	**58.7** (+.2)

4.3 Spatio-temporal Action Detection

We also evaluate ORViT on the task of spatio-temporal action detection on the AVA dataset. In the literature, the action detection task on AVA is often formulated as a two stage prediction procedure. The first step is the detection of bounding boxes, which are often obtained through an off-the-shelf pretrained person detector. The second step involves predicting the action being performed at each detected bounding box. The performance is benchmarked on the end result of these steps and is measured through the Mean Average Precision (MAP) metric. Typically, for fair comparison, the detected person boxes are kept identical across approaches and hence, the final performance depends directly on the ability of the approach to utilize the video and box information.

This task presents an ideal benchmark for evaluating the benefit of ORViT, since all baselines as well as our model operate on identical detected boxes. Table 4 reports the results and shows that ORViT-MViT achieves a significant +1.1 MAP improvement on the MViT-B 16 × 4, thereby showcasing the power of our proposed object-centric representation fusion scheme.

Model	Pretrain	mAP	Param
SlowFast, 4 × 16, R50	K400	21.9	33.7
SlowFast, 8 × 8, R101	K400	23.8	53.0
MViT-B, 16 × 4	K400	25.5	36.4
ORViT MViT-B (Ours)	K400	**26.6**	49.8

Figure 4: Evaluation on AVA.

4.4 Ablations

We perform a comprehensive ablation study on the compositional split of SomethingElse to test the contribution of the different ORViT components (Table 3; for additional ablations see section C in supplementary). We use Mformer as the baseline architecture for ORViT, and use GT boxes.

Components of the ORViT model. We consider the following versions of our model. (i) Single ORViT block (no trajectory stream[5]). We first consider a single application of the ORViT block, but without the trajectory stream. We also compare different video transformer layers at which to apply our ORViT block (namely, the video transformer layer from which to extract the RoI descriptors). We refer to models applied at layer X as *ORViT[L:X]*. (ii) Single ORViT block (with trajectory stream). Here we augment the single ORViT block, with the trajectory stream. We refer to these models as *ORViT[L:X]+Traj*. (iii) Multiple ORViT blocks (with trajectory stream). This is the version of ORViT used in all our experiments. It applies the ORViT block at multiple layers. We chose layers 2,7 and 11 of the video transformer model to apply ORViT block at. All the ablations

[5] We refer to the Object-Dynamics Module as trajectory stream.

Table 3: **Albations.** We report top-1 and top-5 action accuracy on the SomethingElse split. We show (a) Contribution of ORViT components (with parameters number in 10^6 and GFLOPS in 10^9). (b) Other Object-centric baselines , and (c) ORViT with box input not provided by a detector.

(a) **Components**

Layers	Top-1	Top-5	GFLOP	Param
Mformer	60.2	85.8	370	109
ORViT [L:12]	63.9	87.6	373	109
ORViT [L:2]	66.7	89.2	373	109
ORViT [L:2] + Traj	68.8	90.5	381	122
ORViT [L:2, 7, 11] + Traj	**69.7**	**91.0**	405	148

(b) **Object-centric baselines**

Model	Top-1	Top-5
Mformer	60.2	85.8
Mformer + RoIAlign	59.6	84.5
Mformer + Boxes	63.7	86.9
ORViT (Ours)	**69.7**	**91.0**

(c) **Boxes**

Model	Top-1	Top-5
Full boxes	60.9	84.5
Null boxes	60.4	84.2
Grid boxes	60.9	84.8
Random boxes	60.7	85.0
Object Regions (Ours)	**69.7**	**91.0**

were performed on the compositional split in SomethingElse. In the ablation table, we refer to this as *ORViT[L:2,7,11]+Traj*. In the rest of the experiments this is simply referred to as ORViT.

The results are shown in Table 3a. It can be seen that a single layer version of the model already results in considerable improvement (66.7%), and that it is very important to apply it in the earlier layers of video transformer rather than at the end. This is in contrast to the current practice in object-centric approaches (e.g., STRG and STIN) that extract RoIs from the final layer. It can also be seen that the trajectory component improves performance (by 2.1% from 66.7% to 68.8%). Finally, multiple applications of the layer further improve performance to 69.7%.

Object-Centric Baselines. ORViT proposes a way to integrate *object region information* into a transformer architecture. Here we consider two other candidate models to achieve this goal. (i) *Mformer+RoIAlign* uses RoIAlign over the last video transformer layer to extract object features. Then, it concatenates the CLS token with max-pooled object features to classify the action using an MLP. This captures the appearance of objects with global context but without fusing the object-centric information back to the network as we do. (ii) *Mformer+Boxes* uses coordinates and patch tokens. We use the CLS token from the last layer of Mformer, concatenated with trajectory embeddings. To obtain trajectory embeddings, we use a standard self-attention over the coordinates similar to our “Object-Dynamics Module”. This captures the trajectory information with global context without fusing the object-centric information back to the network as we do. The results are shown in Table 3b. *Mformer+RoIAlign* does not improve over the baseline, while *Mformer+Boxes* improves by 3.5%, which it still far from ORViT (69.7%).

How important are the object bounding boxes? Since ORViT changes the architecture of the base video transformer model, we want to check whether the bounding boxes are indeed the source of improvement. We consider several variations where the object bounding boxes are replaced with other values. (i) *All boxes*: all boxes are given the coordinates of the entire image ($[0, 0, 1, 1]$). (ii) *Null boxes*: all boxes are initialized to zeros. (iii) *Grid boxes*: each of the 4 bounding boxes is one fourth of the image. (iv) *Random boxes* - each box is chosen uniformly at random. See Table 3c for results. We observe a large drop in performance for all these baselines, which confirms the important role of the object regions in ORViT. Finally, we ask whether tracking information is important, as opposed to just detection. Namely, what happens if we keep the bounding boxes, but shuffle their allocation to objects. We find that this results in degradation from 69.7 to 68.2, indicating that the model can perform relatively well with only detection information.

Box Position Encoder. Our “Box Position Encoder” transforms from a tensor of size TO to size THW. We compare between our implementation that uses boxes to a simpler one that expands the shape of TO to THW without using boxes. Our approach obtains 69.7 compared to 68.4, showing that our box-based encoding performs better. For more details, see Section C in the Supplementary.

5 Conclusions

Objects are a key element of human visual perception, but their modeling is still a challenge for machine vision. The ORViT model we present here makes use of simple object detection information to generate a contextualized representation of the entire scene. We note that such integration is particularly natural in transformer models, where an object region has the same role in the architecture as the uniformly-spaced patch tokens. In our current implementation, we use externally provided boxes. However, it will be interesting to replace the externally provided boxes with boxes that the model generates itself without supervision. We leave this challenge to future work.

REFERENCES

Humam Alwassel, Fabian Caba Heilbron, and Bernard Ghanem. Action search: Spotting targets in videos and its application to temporal action localization. In *The European Conference on Computer Vision (ECCV)*, September 2018.

A. Arnab, Chen Sun, and C. Schmid. Unified graph structured models for video understanding. *ArXiv*, abs/2103.15662, 2021a.

Anurag Arnab, Mostafa Dehghani, Georg Heigold, Chen Sun, Mario Lučić, and Cordelia Schmid. Vivit: A video vision transformer, 2021b.

Amir Bar, Roei Herzig, Xiaolong Wang, Anna Rohrbach, Gal Chechik, Trevor Darrell, and A. Globerson. Compositional video synthesis with action graphs. In *ICML*, 2021.

Fabien Baradel, Natalia Neverova, Christian Wolf, Julien Mille, and Greg Mori. Object level visual reasoning in videos. In *European Conf. Comput. Vision*, pp. 105–121, 2018.

Peter W Battaglia, Jessica B Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, et al. Relational inductive biases, deep learning, and graph networks. *arXiv preprint arXiv:1806.01261*, 2018.

Gedas Bertasius, Heng Wang, and Lorenzo Torresani. Is space-time attention all you need for video understanding? In *Proceedings of the International Conference on Machine Learning (ICML)*, July 2021.

Alex Bewley, Zongyuan Ge, Lionel Ott, Fabio Ramos, and Ben Upcroft. Simple online and realtime tracking. In *2016 IEEE International Conference on Image Processing (ICIP)*, pp. 3464–3468, 2016. doi: 10.1109/ICIP.2016.7533003.

S. Buch, V. Escorcia, C. Shen, B. Ghanem, and J. C. Niebles. Sst: Single-stream temporal action proposals. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6373–6382, 2017.

Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm (eds.), *Computer Vision – ECCV 2020*, pp. 213–229, 2020.

Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6299–6308, 2017.

Linda L Chao and Alex Martin. Representation of manipulable man-made objects in the dorsal stream. *NeuroImage*, 12:478–484, 2000.

Yu-Wei Chao, Sudheendra Vijayanarasimhan, Bryan Seybold, David A. Ross, Jia Deng, and Rahul Sukthankar. Rethinking the faster R-CNN architecture for temporal action localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. Uniter: Universal image-text representation learning. In *ECCV*, 2020.

Ekin Dogus Cubuk, Barret Zoph, Jon Shlens, and Quoc Le. Randaugment: Practical automated data augmentation with a reduced search space. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 18613–18624. Curran Associates, Inc., 2020. URL `https://proceedings.neurips.cc/paper/2020/file/d85b63ef0ccb114d0a3bb7b7d808028f-Paper.pdf`.

Xiyang Dai, Bharat Singh, Guyue Zhang, Larry S. Davis, and Yan Qiu Chen. Temporal context network for activity localization in videos. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

Dima Damen, Hazel Doughty, Giovanni Maria Farinella, , Antonino Furnari, Jian Ma, Evangelos Kazakos, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, and Michael Wray. Rescaling egocentric vision. *CoRR*, abs/2006.13256, 2020.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

Jeffrey Donahue, Lisa Anne Hendricks, Sergio Guadarrama, Marcus Rohrbach, Subhashini Venugopalan, Kate Saenko, and Trevor Darrell. Long-term recurrent convolutional networks for visual recognition and description. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2625–2634, 2015.

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

Alexei A Efros, Alexander C Berg, Greg Mori, and Jitendra Malik. Recognizing action at a distance. In *Computer Vision, IEEE International Conference on*, volume 3, pp. 726–726. IEEE Computer Society, 2003.

C. Feichtenhofer, H. Fan, J. Malik, and K. He. Slowfast networks for video recognition. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 6201–6210, 2019. doi: 10.1109/ICCV.2019.00630.

Christoph Feichtenhofer, Axel Pinz, and Andrew Zisserman. Convolutional two-stream network fusion for video action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1933–1941, 2016.

Vittorio Gallese, Luciano Fadiga, Leonardo Fogassi, and Giacomo Rizzolatti. Action recognition in the premotor cortex. *Brain : a journal of neurology*, 119 (Pt 2):593–609, 1996.

Chen Gao, Jiarui Xu, Yuliang Zou, and Jia-Bin Huang. Drg: Dual relation graph for human-object interaction detection. *ArXiv*, abs/2008.11714, 2020.

Jiyang Gao, Zhenheng Yang, Kan Chen, Chen Sun, and Ram Nevatia. Turn tap: Temporal unit regression network for temporal action proposals. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

Rohit Girdhar, Joao Carreira, Carl Doersch, and Andrew Zisserman. Video action transformer network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 244–253, 2019.

Raghav Goyal, Samira Ebrahimi Kahou, Vincent Michalski, Joanna Materzynska, Susanne Westphal, Heuna Kim, Valentin Haenel, Ingo Fruend, Peter Yianilos, Moritz Mueller-Freitag, et al. The" something something" video database for learning and evaluating visual common sense. In *ICCV*, pp. 5, 2017.

Chunhui Gu, Chen Sun, David A. Ross, Carl Vondrick, Caroline Pantofaru, Yeqing Li, Sudheendra Vijayanarasimhan, George Toderici, Susanna Ricco, Rahul Sukthankar, Cordelia Schmid, and Jitendra Malik. AVA: A video dataset of spatio-temporally localized atomic visual actions. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pp. 6047–6056. IEEE Computer Society, 2018a.

Chunhui Gu, Chen Sun, David A Ross, Carl Vondrick, Caroline Pantofaru, Yeqing Li, Sudheendra Vijayanarasimhan, George Toderici, Susanna Ricco, Rahul Sukthankar, et al. Ava: A video dataset of spatio-temporally localized atomic visual actions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6047–6056, 2018b.

Abhinav Gupta and Larry S. Davis. Objects in action: An approach for combining action understanding and object perception. *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8, 2007.

P. Hacker. Events and objects in space and time. *Mind*, pp. 1–19, 1982.

Fan Haoqi, Xiong Bo, Mangalam Karttikeya, Li Yanghao, Yan Zhicheng, Malik Jitendra, and Feichtenhofer Christoph. Multiscale vision transformers. *arXiv:2104.11227*, 2021.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proc. Conf. Comput. Vision Pattern Recognition*, pp. 770–778, 2016.

Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 2980–2988, 2017.

Hannah B. Helbig, Markus Graf, and Markus Kiefer. The role of action representations in visual object recognition. *Experimental Brain Research*, 174:221–228, 2006.

Roei Herzig, Moshiko Raboh, Gal Chechik, Jonathan Berant, and Amir Globerson. Mapping images to scene graphs with permutation-invariant structured prediction. In *Advances in Neural Information Processing Systems (NIPS)*, 2018.

Roei Herzig, Elad Levi, Huijuan Xu, Hang Gao, Eli Brosh, Xiaolong Wang, Amir Globerson, and Trevor Darrell. Spatio-temporal action graph networks. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pp. 0–0, 2019.

Roei Herzig, Amir Bar, Huijuan Xu, Gal Chechik, Trevor Darrell, and Amir Globerson. Learning canonical representations for scene graph to image generation. In *European Conference on Computer Vision*, 2020.

Geoffrey E. Hinton, Nitish Srivastava, A. Krizhevsky, Ilya Sutskever, and R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *ArXiv*, abs/1207.0580, 2012.

Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q. Weinberger. Deep networks with stochastic depth. In *ECCV*, 2016.

Max Jaderberg, K. Simonyan, Andrew Zisserman, and K. Kavukcuoglu. Spatial transformer networks. In *NIPS*, 2015.

M. Jain, A. Ghodrati, and C. G. M. Snoek. Actionbytes: Learning from trimmed videos to localize actions. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1168–1177, 2020.

Achiya Jerbi, Roei Herzig, Jonathan Berant, Gal Chechik, and Amir Globerson. Learning object detection from captions via textual scene attributes. *ArXiv*, abs/2009.14558, 2020.

Shuiwang Ji, Wei Xu, Ming Yang, and Kai Yu. 3d convolutional neural networks for human action recognition. *IEEE transactions on pattern analysis and machine intelligence*, 35(1):221–231, 2013.

Justin Johnson, Agrim Gupta, and Li Fei-Fei. Image generation from scene graphs. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1219–1228, 2018.

R. Kálmán. A new approach to linear filtering and prediction problems” transaction of the asme journal of basic. In *transaction of the asme journal of basic*, 1960.

Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. Large-scale video classification with convolutional neural networks. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 1725–1732, 2014.

Keizo Kato, Yin Li, and Abhinav Gupta. Compositional learning for human object interaction. In *ECCV*, 2018.

Will Kay, Joao Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, et al. The kinetics human action video dataset. *arXiv preprint arXiv:1705.06950*, 2017.

Dahun Kim, Sanghyun Woo, Joon-Young Lee, and In So Kweon. Deep video inpainting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5792–5801, 2019.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

Ranjay Krishna, Ines Chami, Michael S. Bernstein, and Li Fei-Fei. Referring relationships. *ECCV*, 2018.

H. Kuhn. The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2:83–97, 1955.

C. Lea, M. D. Flynn, R. Vidal, A. Reiter, and G. D. Hager. Temporal convolutional networks for action segmentation and detection. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1003–1012, 2017.

Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. Visualbert: A simple and performant baseline for vision and language. *ArXiv*, abs/1908.03557, 2019.

Xiujun Li, Xi Yin, Chunyuan Li, Xiaowei Hu, Pengchuan Zhang, Lei Zhang, Lijuan Wang, Houdong Hu, Li Dong, Furu Wei, Yejin Choi, and Jianfeng Gao. Oscar: Object-semantics aligned pre-training for vision-language tasks. *ECCV 2020*, 2020.

Yingwei Li, Yi Li, and Nuno Vasconcelos. Resound: Towards action recognition without representation bias. In *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.

Ji Lin, Chuang Gan, and Song Han. Tsm: Temporal shift module for efficient video understanding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019a.

Tianwei Lin, Xu Zhao, Haisheng Su, Chongjing Wang, and Ming Yang. Bsn: Boundary sensitive network for temporal action proposal generation. In Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss (eds.), *Computer Vision – ECCV 2018*, 2018.

Tianwei Lin, Xiao Liu, Xin Li, Errui Ding, and Shilei Wen. BMN: boundary-matching network for temporal action proposal generation. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV*, pp. 3888–3897. IEEE, 2019b. URL `https://doi.org/10.1109/ICCV.2019.00399`.

Tsung-Yi Lin, M. Maire, Serge J. Belongie, James Hays, P. Perona, D. Ramanan, Piotr Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014.

Tsung-Yi Lin, Piotr Dollár, Ross B. Girshick, Kaiming He, Bharath Hariharan, and Serge J. Belongie. Feature pyramid networks for object detection. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 936–944, 2017.

Fuchen Long, Ting Yao, Zhaofan Qiu, Xinmei Tian, Jiebo Luo, and Tao Mei. Gaussian temporal awareness networks for action localization. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pp. 344–353. Computer Vision Foundation / IEEE, 2019.

S. Ma, L. Sigal, and S. Sclaroff. Learning activity progression in lstms for activity detection and early detection. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1942–1950, 2016.

Joanna Materzynska, Tete Xiao, Roei Herzig, Huijuan Xu, Xiaolong Wang, and Trevor Darrell. Something-else: Compositional action recognition with spatial-temporal interaction networks. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2020.

Megha Nawhal and Greg Mori. Activity graph transformer for temporal action localization, 2021.

Mandela Patrick, Dylan Campbell, Yuki M. Asano, Ishan Misra Florian Metze, Christoph Feichtenhofer, Andrea Vedaldi, and Joao F. Henriques. Keeping your eye on the ball: Trajectory attention in video transformers, 2021.

A. Quinton. Objects and events. *Mind*, pp. 197–214, 1979.

Moshiko Raboh, Roei Herzig, Gal Chechik, Jonathan Berant, and Amir Globerson. Differentiable scene graphs. In *Winter Conf. on App. of Comput. Vision*, 2020.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015. URL https://proceedings.neurips.cc/paper/2015/file/14bfa6bb14875e45bba028a21ed38046-Paper.pdf.

A. Richard and J. Gall. Temporal action detection using a statistical language model. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3131–3140, 2016.

Kate Saenko, Ben Packer, Chao-Yeh Chen, Sunil Bandla, Yong Jae Lee, Yangqing Jia, Juan Carlos Niebles, Daphne Koller, Li Fei-Fei, Kristen Grauman, and Trevor Darrell. Mid-level features improve recognition of interactive activities. *arXiv preprint arXiv:1912.06992*, 2012.

Adam Santoro, David Raposo, David G Barrett, Mateusz Malinowski, Razvan Pascanu, Peter Battaglia, and Timothy Lillicrap. A simple neural network module for relational reasoning. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

Baifeng Shi, Qi Dai, Yadong Mu, and Jingdong Wang. Weakly-supervised action localization by generative attention modeling. *arXiv preprint arXiv:2003.12424*, 2020.

Zheng Shou, Dongang Wang, and Shih-Fu Chang. Temporal action localization in untrimmed videos via multi-stage cnns. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pp. 1049–1058. IEEE Computer Society, 2016.

Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. In *Advances in neural information processing systems*, pp. 568–576, 2014.

Bharat Singh, Tim K. Marks, Michael Jones, Oncel Tuzel, and Ming Shao. A multi-stream bi-directional recurrent neural network for fine-grained action detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

Chen Sun, Abhinav Shrivastava, Carl Vondrick, Kevin Murphy, Rahul Sukthankar, and Cordelia Schmid. Actor-centric relation network. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 318–334, 2018.

Hao Hao Tan and Mohit Bansal. Lxmert: Learning cross-modality encoder representations from transformers. In *EMNLP*, 2019.

Graham W Taylor, Rob Fergus, Yann LeCun, and Christoph Bregler. Convolutional learning of spatio-temporal features. In *European conference on computer vision*, pp. 140–153. Springer, 2010.

Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning*, pp. 10347–10357. PMLR, 2021.

Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pp. 4489–4497, 2015.

Gül Varol, Ivan Laptev, and Cordelia Schmid. Long-term temporal convolutions for action recognition. *IEEE transactions on pattern analysis and machine intelligence*, 40(6):1510–1517, 2018.

Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Van Gool. Temporal segment networks for action recognition in videos. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(11):2740–2755, 2019.

Xiaolong Wang and Abhinav Gupta. Videos as space-time region graphs. In *ECCV*, 2018.

Chao-Yuan Wu, Christoph Feichtenhofer, Haoqi Fan, Kaiming He, Philipp Krähenbühl, and Ross Girshick. Long-Term Feature Banks for Detailed Video Understanding. In *CVPR*, 2019a.

Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. `https://github.com/facebookresearch/detectron2`, 2019b.

Bingjie Xu, Yongkang Wong, Junnan Li, Qi Zhao, and M. Kankanhalli. Learning to detect human-object interactions with knowledge. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2019–2028, 2019.

H. Xu, A. Das, and K. Saenko. R-c3d: Region convolutional 3d network for temporal activity detection. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 5794–5803, 2017.

Keyulu Xu, Jingling Li, Mozhi Zhang, Simon S. Du, Ken ichi Kawarabayashi, and Stefanie Jegelka. What can neural networks reason about? In *International Conference on Learning Representations*, 2020. URL `https://openreview.net/forum?id=rJxbJeHFPS`.

Serena Yeung, Olga Russakovsky, Greg Mori, and Li Fei-Fei. End-to-end learning of action detection from frame glimpses in videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. Graph contrastive learning with augmentations. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 5812–5823. Curran Associates, Inc., 2020.

Yuan Yuan, Yueming Lyu, Xi Shen, Ivor W. Tsang, and Dit-Yan Yeung. Marginalized average attentional network for weakly-supervised learning. In *International Conference on Learning Representations*, 2019.

Zehuan Yuan, Jonathan C. Stroud, Tong Lu, and Jia Deng. Temporal action localization by structured maximal sums. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

Joe Yue-Hei Ng, Matthew Hausknecht, Sudheendra Vijayanarasimhan, Oriol Vinyals, Rajat Monga, and George Toderici. Beyond short snippets: Deep networks for video classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4694–4702, 2015.

Vinicius Zambaldi, David Raposo, Adam Santoro, Victor Bapst, Yujia Li, Igor Babuschkin, Karl Tuyls, David Reichert, Timothy Lillicrap, Edward Lockhart, et al. Relational deep reinforcement learning. *arXiv preprint arXiv:1806.01830*, 2018.

Runhao Zeng, Wenbing Huang, Mingkui Tan, Yu Rong, Peilin Zhao, Junzhou Huang, and Chuang Gan. Graph convolutional networks for temporal action localization. In *ICCV*, 2019.

Chuhan Zhang, Ankush Gputa, and Andrew Zisserman. Temporal query networks for fine-grained video understanding. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

Yue Zhao, Yuanjun Xiong, Limin Wang, Zhirong Wu, Xiaoou Tang, and Dahua Lin. Temporal action detection with structured segment networks. In *ICCV*, 2017.

Bolei Zhou, Alex Andonian, Aude Oliva, and Antonio Torralba. Temporal relational reasoning in videos. *European Conference on Computer Vision*, 2018.

Dimitri Zhukov, Jean-Baptiste Alayrac, Ramazan Gokberk Cinbis, David Fouhey, Ivan Laptev, and Josef Sivic. Cross-task weakly supervised learning from instructional videos. In *Proc. Conf. Comput. Vision Pattern Recognition*, 2019.

In this supplementary file, we provide additional information about our model, implementation details, experimental results, and qualitative examples. Specifically, Section A provides additional implementation details, Section B provides additional model details, Section C provides additional ablations of our approach, Section D provides more experiment results, and we show qualitative visualizations to demonstrate our approach in Section E.

A Additional Implementation Details

We add ORViT to the existing transformer models MFormer, MViT and TimesFormer. These are all implemented based on the SlowFast (Feichtenhofer et al., 2019) library (available at `https://github.com/facebookresearch/SlowFast`) and we base our code on that library. Next, we elaborate on how we extract the object regions, and for each dataset, we add additional implementation details.

A.1 Detector and Tracker

Detector. For SSv2 and EK100, we used the supplied detection boxes from the original codebase. In SSv2, the detection class categories are grouped into *hand* and *object* involved in action. For Diving48, we used off-the-shelf Faster R-CNN detector (Ren et al., 2015; He et al., 2017) with ResNet-50 backbone (He et al., 2016) and Feature Pyramid Network (FPN) (Lin et al., 2017) that is pre-trained on the MS COCO (Lin et al., 2014) dataset. We used the Detectron2 (Wu et al., 2019b) implementation. For AVA, we used the supplied detection boxes for the spatio-temporal action detection task that were first obtained by Faster-RCNN pre-trained over MS COCO and then fine-tuned on AVA as in (Wu et al., 2019a). We set the number of objects in our model to 4 in SSv2 and EK100, 6 in AVA, and 10 in Diving48. If fewer objects are presented, we set the object coordinates with a zero vector.

Tracker. Once we have the detector results, we apply multi-object tracking to find correspondence between the objects in different frames. We use SORT (Bewley et al., 2016): a simple tracker implemented based on Kalman Filter (Kálmán, 1960) and the Hungarian matching algorithm (KM) (Kuhn, 1955). At each step, the Kalman Filter predicts plausible instances in the current frame based on previous tracks. Next, the predictions are matched with single-frame detections by the Hungarian matching algorithm. It is important to note that the tracker does not require any training and does not use any additional data. If an object does not appear in one of the frames, we set the coordinates in these frames to zeros.

A.2 Something-Something v2

Dataset.. SSv2 (Goyal et al., 2017) contains 174 action categories of common human-object interactions. We follow the official splits from (Materzynska et al., 2020) to train and test our model for action recognition, compositional action recognition, and few-shot.

Optimization details. For the standard SSv2 (Materzynska et al., 2020) dataset, we trained 16 frames with sample rate 4 and batch-size 48 on 8 RTX 3090 GPUs. We train our network for 35 epochs with Adam optimizer (Kingma & Ba, 2014) with a momentum of 0.9 and Gamma 0.1. Following (Patrick et al., 2021), we use $lr = 5e - 5$ with $\times 10$ decay steps at epochs $0, 20, 30$. Additionally, we used Automatic Mixed Precision, which is implemented by PyTorch. We initialize from a Kinetics-400 pre-trained model (Kay et al., 2017). For the *ORViT Mformer-L* model, we fine-tuned from the SSv2 pre-trained model provided by (Patrick et al., 2021) and train with 32 frames. The optimization policy is similar to the above, except we used a different learning rate: $1e - 5$ for the pre-trained parameters, and $1e - 4$ for the ORViT parameters.

For the compositional action recognition task, we trained on the SomethingElse splits (Materzynska et al., 2020). We train with a batch size of 32 and a learning rate of $3e - 5$.

Regularization details. We use weight decay of 0.05, a dropout (Hinton et al., 2012) of 0.5 before the final classification, dropout of 0.3 after the ORViT block, and DropConnect (Huang et al., 2016) with rate 0.2.

Training details. We use a standard crop size of 224, and we jitter the scales from 256 to 320. Additionally, we use RandAugment (Cubuk et al., 2020) with maximum magnitude 20 for each frame separately.

Inference details. We take 3 spatial crops per single clip to form predictions over a single video in testing as done in Patrick et al. (2021).

A.3 EPICKITCHENS100

Dataset. EK100 (Damen et al., 2020) contains 700 egocentric videos of daily kitchen activities. This dataset includes 300 noun and 97 verb classes, and we report verb, noun, and action top-1 accuracy, while the highest-scoring of the verb and noun pairs constitutes the action label.

Optimization details. We trained over videos of 16 frames with sample rate 4 and batch-size 16 on 8 Quadro RTX 8000 GPUs. We train our network for 35 epochs with Adam optimizer (Kingma & Ba, 2014) with a momentum of 0.9 and Gamma 0.1. Following (Patrick et al., 2021), we use $lr = 5e-5$ with $\times 10$ decay steps at epochs $0, 30, 40$. Additionally, we used Automatic Mixed Precision, which is implemented by PyTorch. We initialize from a Kinetics-400 pre-trained model (Kay et al., 2017).

Training details. We use crop size of 336 for the *ORViT Mformer-HR*. We jitter the scales from 384 to 480. Additionally, we use RandAugment (Cubuk et al., 2020) with maximum magnitude 20.

Inference details. We take 3 spatial crops with 10 different clips sampled randomly to aggregate predictions over a single video in testing.

A.4 DIVING48

Dataset. Diving48 (Li et al., 2018) contains 16K training and 3K testing videos spanning 48 fine-grained diving categories of diving activities. For all of these datasets, we use standard classification accuracy as our main performance metric.

Optimization details. We trained over videos of 32 frames with sample rate 8 and batch-size 8 on 8 Quadro RTX 8000 GPUs. We train our network for 35 epochs with Adam optimizer (Kingma & Ba, 2014) with a momentum of 0.9 and Gamma 0.1. We use $lr = 3.75e - 5$ with $\times 10$ decay steps at epochs $0, 20, 30$. Additionally, we used Automatic Mixed Precision, which is implemented by PyTorch. We initialize from a Kinetics-400 pre-trained model (Kay et al., 2017).

Training details. We use a standard crop size of 224 for the standard model and jitter the scales from 256 to 320. Additionally, we use RandomFlip augmentation. Finally, we sampled the T frames from the start and end diving annotation time, followed by (Zhang et al., 2021).

Inference details. We take 3 spatial crops per single clip to form predictions over a single video in testing same as in Bertasius et al. (2021).

Object-Dynamics Module. As we show in Table 4d, we compared different self-attention mechanisms, and the standard self-attention usually performed better. However, we observe a slight improvement when we perform a trajectory self-attention (Patrick et al., 2021) instead of the standard self-attention.

A.5 AVA

Architecture. SlowFast (Feichtenhofer et al., 2019) and MViT (Haoqi et al., 2021) are using a detection architecture with a RoI Align head on top of the spatio-temporal features. We followed their implementation to allow a direct comparison. Next we elaborate on the RoI Align head proposed in SlowFast (Feichtenhofer et al., 2019). First, we extract the feature maps from our ORViT MViT model by using the RoIAlign layer. Next, we take the 2D proposal at a frame into a 3D RoI by replicating it along the temporal axis, followed by a temporal global average pooling. Then, we max-pooled the RoI features and fed them to an MLP classifier for prediction.

Optimization details. To allow a direct comparison, we used the same configuration as in MViT (Haoqi et al., 2021). We trained 16 frames with sample rate 4, depth of 16 layers and batch-size 32 on 8 RTX 3090 GPUs. We train our network for 30 epochs with an SGD optimizer. We use

$lr = 0.03$ with a weight decay of $1e-8$ and a half-period cosine schedule of learning rate decaying. We use mixed precision and fine-tune from an MViT-B, 16×4 pre-trained model.

Training details. We use a standard crop size of 224 and we jitter the scales from 256 to 320. We use the same ground-truth boxes and proposals that overlap with ground-truth boxes by $IoU > 0.9$ as in (Feichtenhofer et al., 2019).

Inference details. We perform inference on a single clip with 16 frames. For each sample, the evaluation frame is centered in frame 8. We use a crop size of 224 in test time. We take 1 spatial crop with 10 different clips sampled randomly to aggregate predictions over a single video in testing.

A.6 Few Shot Compositional Action Recognition

We also evaluate on the few-shot compositional action recognition task in (Materzynska et al., 2020). For this setting, we have 88 *base* action categories and 86 *novel* action categories. We train on the base categories (113K/12K for training/validation) and finetune on few-shot samples from the novel categories (for 5-shot, 430/50K for training/validation; for 10-shot, 860/44K for training/validation).

B Additional Model details

B.1 Object-Region Attention

As explained in section 3.2, there are two inputs to the ORViT block. The first is the output of the preceding transformer block, represented as a set of spatio-temporal tokens $X \in \mathbb{R}^{THW \times d}$. The second input is a set of bounding boxes for objects across time, provided by an off-the-shelf object detector or available directly in the dataset. These bounding boxes are denoted by $B \in \mathbb{R}^{TO \times 4}$.

Object-Region Attention. Given the patch token features X and the boxes B, we use RoIAlign (He et al., 2017) layer, which uses the patch tokens X and box coordinates B to obtain object region crops. This is followed by max-pooling and an MLP. To these features we add a learnable object-time position encoding $\mathcal{P} \in \mathbb{R}^{TO \times d}$ to encode the positional object information. We also use a coordinate embedding by applying an MLP on the boxes coordinates, resulting in $\mathcal{L} \in \mathbb{R}^{d}$:

$$\mathcal{L} := \text{MLP}(B) \tag{4}$$

where $B \in \mathbb{R}^{T \times O \times d}$ is the boxes coordinates. This leads to an improved object features:

$$\mathcal{O} := \text{MLP}(\text{MaxPool}(\text{RoIAlign}(X, B))) + \mathcal{L} + \mathcal{P} \tag{5}$$

where the token features are $X \in \mathbb{R}^{THW \times d}$. We pass these features into the self-attention layers as explained in the “Object-Region attention” subsection.

C Additional Ablations

We perform an ablation study of each of the components in Table 4 to show the effectiveness of the different components of our model. All ablations are on the SomethingElse dataset and use Mformer as the baseline architecture for ORViT unless stated otherwise.

Contribution of Appearance and motion Streams. In Table 4a, we show the “Object-Region Attention” is an important factor for the improvement, responsible for a 7.2% gain from the baseline without significant parameters addition (2M parameters compared to the baseline). This highlights our contribution that object interactions are indeed crucial for video transformers. Additionally, adding trajectory information with coordinates in the “Object-Dynamics Module” improved by another 2.3%.

ORViT Blocks. In Table 4b, we show which layers are most important for adding the ORViT block. The experiments show that adding this information at the network’s beginning, middle, and the end is the most effective (layer 2, 7, 11). We also note that, surprisingly, adding ORViT at the second layer has the most significant contribution (8.6% improvements).

Different self attention in “Object-Region Attention”. In Table 4c, we compared different self-attention mechanisms: joint space-time, divided space-time, and trajectory attention to the *Mformer*

Table 4: **Albations.** Evaluation of different model ablations and baselines on the "SomethingElse" split (Tables (a-d) see text). We report pretrain, param (10^6), GFLOPS (10^9), and top-1 and top-5 video action recognition accuracy. Table (e) reports ablations on the Diving dataset.

(a) **Streams**

Model	Top-1	Top-5	GFLOPs(10^9)	Param(10^6)
Baseline	60.2	85.5	369.5	109
+ Object-Region Att.	67.4	89.8	382.3	111
+ Object-Motion Mod.	**69.7**	**91.0**	405	148

(b) **Blocks Ablation**

Layers	Top-1	Top-5	GFLOPs (10^9)	Param (10^6)
Baseline	60.2	85.8	370	109
2	68.8	90.4	381	122
7	68.9	90.5	381	122
11	66.8	89.3	381	122
2, 7	69.3	90.6	393	135
2, 7, 11	**69.7**	**91.0**	405	148

(c) **Object-Region Attention**

Model	Top-1	Top-5
Baseline	60.2	85.8
Ours /w Joint attn.	68.9	90.4
Ours /w Divided attn.	69.3	90.6
Ours /w Trajectory attn.	**69.7**	**91.0**

(d) **Object-Motion Module**

Model	Top-1	Top-5
GCN	67.7	89.8
Trajectory Self-attention	69.4	90.5
Self-attention	**69.7**	**91.0**

(e) **Components**

Layers	Top-1	Param
Baseline	80.0	121
ORViT [:12]	85.4	121
ORViT [:2]	86.8	121
ORViT [:2] + Motion	87.5	135
ORViT [:2, 7, 11] + Motion	**88.0**	163

baseline, which uses trajectory attention in all layers. We observed that trajectory attention is slightly better. However, it can be seen that our object region approach is not sensitive to these choices, indicating that the generic approach is the main reason for the observed improvements.

Processing trajectory information. In Table 4d, we compared our self-attention (see "Object-Dynamics Module" in Section 3.2) with other standard baseline models: GCN (Kipf & Welling, 2016) and trajectory self-attention (Patrick et al., 2021). For the GCN, we use a standard implementation with 2 hidden layers, while for the trajectory attention, we treat the O objects as the spatial dimension. Our model improves the GCN baseline by 2.0%, and the difference between the self-attention layers is significant (0.3% difference).

Components on Diving48. Following our ablations in Table 3a, we also validate our approach on the Diving48 dataset in Table 4e. We used the TimeSformer trained on videos of 32 frames as a baseline for a fair comparison. It can be seen that a single layer version of the model already results in considerable improvement (85.4%) and that it is important to apply it in the earlier layers of transformers than at the end (86.8% compared to 85.4%). Additionally, the "Object-Dynamics Module" improves performance to 87.5%. Finally, multiple applications of the layer further improve performance to 88.0%.

Replacing ORViT with Trajectory Attention. We observe that Joint and Divided self attention layers (Bertasius et al., 2021; Arnab et al., 2021b) have similar results to the Trajectory Attention (Kim et al., 2019), as seen in Table 4c. However, we would like to demonstrate that Trajectory Attention is not the main reason for the improvement. Thus, we replace our ORViT with a standard Trajectory Attention on the Diving48 and AVA datasets. The top1 accuracy on Diving48 are improved by 4.5% (from 80.0 to 84.5) with trajectory attention, while using our *ORViT+TimeSformer* achieves 88.0 (3.5% improvements on top of that). The MAP on AVA are the same as the baseline with trajectory attention (25.5), while using our *ORViT+MViT-B* achieves 26.6 (1.1 improvements on top of the baseline). We note that our Mformer is the baseline on EK100, SSv2, and SomethingElse, and therefore the trajectory attention is already part of the model, and hence this demonstration is not needed.

Box Position Encoder. Our "Box Position Encoder" transforms from a tensor of size TO to size THW. Our implementation of this transformation uses box information so that each object is mapped to the "correct" region in space. A simpler approach would have been to expand the shape of TO to THW without using boxes. We refer to the latter as a standard tensor expansion. Comparing the two methods, we find out that our approach obtains 69.7 compared to 68.4, showing that our box-based encoding performs better.

D Additional Experiments

Here we present additional results of the standard action recognition task. Additionally, we show a lightweight version of the "Object-Dynamics Module" that significantly reduces the model parameters without losing significant performance.

D.1 Light-weight Object-Dynamics Module

Dataset	Model	Boxes	Dimension	Top-1	Top-5	GFLOPs (10^9)	Param (10^6)
SomethingElse	Baseline	-	-	60.2	85.8	369.5	109
	Ours	GT	128	68.7	90.3	382	112
			256	68.9	90.5	383	114
			768	69.7	91.0	408	148
SSv2	Baseline	-	-	66.5	90.1	369.5	109
	Ours	GT	128	73.4	93.5	382	112
			256	73.7	93.6	383	114
			768	73.8	93.7	405	148
		Detected	128	67.2	90.4	382	112
			256	67.3	90.5	383	114
			768	67.9	90.5	405	148

Table 5: **A light-weighted version of ORViT**.

In Table 4a, we show that the "Object-Dynamics Module" improves by 2.3% the top-1 accuracy with an additional 39M parameters (148M Vs. 109M). We would like to demonstrate that our main approach is the main reason for the observed improvements and not necessarily the addition of more parameters. Here, we present a light-weight version of the module that reduces the embedding dimensions without losing significant accuracy. See Table 5.

As mentioned in the main paper (see Section 3), we use $\widetilde{B}$ for the coordinate embeddings and object-time position embedding in the "Object-Dynamics Module". We validate how the dimension size of $\widetilde{B}$ is responsible for the action accuracy. We observe that reducing the dimension of $\widetilde{B}$ from 768 to 256 has little impact on the action accuracy in SSv2 (73.8% Vs. 73.7% for ground-truth boxes and 67.9% Vs. 67.3% for detected boxes) and SomethingElse (68.7% Vs. 67.9%), although having only 114M model parameters (only addition of 5M parameters to the Mformer baseline that has 109M).

Indeed this indicates that our main approach is the main reason for the observed improvements and not necessarily more parameters.

D.2 Results

We next report in Table 6 the full results table for the standard action recognition task, including extra models, which were not included in the main paper.

E Qualitative Visualizations

To provide insight into the inner representation of ORViT we provide further visualization next. See Figure 5 and Figure 6. In Figure 5, we visualize the attention map of the CLS token on all spatial tokens. It can be seen that object-regions indeed affect these spatial maps. For example, "Tearing something into two pieces" (top left corner) demonstrates that *ORViT+Mformer* successfully separates the two pieces of the paper, while the *Mformer* baseline does not. Next, in Figure 6 we visualize the attention allocated to each of the object keys. It can be seen that the object keys in *ORViT* indeed affect their corresponding spatial tokens.

Table 6: **Comparison to the state-of-the-art on video action recognition.** We report pretrain, param (10^6), GFLOPS (10^9) and top-1 (%) and top-5 (%) video action recognition accuracy on SSv2. We also report when our model uses ground-truth (GT) and detected boxes. On Epic-Kitchens100 (EK100), we report top-1 (%) action (A), verb (V), and noun (N) accuracy. On Diving48 we report pretrain, number of frames and top-1 (%) video action recognition accuracy. For EK100 and Diving48 we used only detected boxes since ground-truth does not exist.

(a) **Something–Something V2**

Model	Boxes	Pretrain	Top-1	Top-5	GFLOPs×views (10^9)	Param (10^6)
SlowFast, R50	✗	K400	61.7	87.0	65.7×3×1	34.1
SlowFast, R101	✗	K400	63.1	87.6	106×3×1	53.3
TSM	✗	K400	63.4	88.5	62.4×3×2	42.9
STM	✗	IN-1K	64.2	89.8	66.5×3×10	-
MSNet	✗	IN-1K	64.7	89.4	67×1×1	24.6
TEA	✗	IN-1K	65.1	-	70×3×10	-
bLVNet	✗	IN-1K	65.2	90.3	128.6×3×10	-
VidTr-L	✗	IN-21K+K400	60.2	-	351×3×10	-
TimeSformer-L	✗	IN-21K	62.5	-	1703×3×1	121.4
ViViT-L	✗	IN-21K+K400	65.4	89.8	3992×4×3	-
MViT-B, 32	✗	K400	67.1	90.8	170×3×1	36.6
MViT-B, 64	✗	K400	67.7	90.9	455×3×1	36.6
MViT-B, 32	✗	K600	67.8	91.3	170×3×1	36.6
MViT-B, 64	✗	K600	68.7	91.5	236×3×1	53.2
Mformer	✗	IN-21K+K400	66.5	90.1	369.5 × 3 × 1	-
Mformer-L	✗	IN-21K+K400	68.1	91.2	1185.1 × 3 × 1	109
Mformer + STIN	GT	IN-21K+K400	66.9	89.4	369.5 × 3 × 1	111
Mformer + STRG	GT	IN-21K+K400	69.1	90.9	375 × 3 × 1	117
Mformer + STRG + STIN	GT	IN-21K+K400	69.2	90.9	375 × 3 × 1	119
ORViT Mformer (Ours)	Detected	IN-21K+K400	**67.9** (+1.4)	90.5 (+0.4)	405 × 3 × 1	148
ORViT Mformer (Ours)	GT	IN-21K+K400	**73.8** (+7.3)	**93.6** (+3.5)	405 × 3 × 1	148
ORViT Mformer-L (Ours)	Detected	IN-21K+K400	**69.5** (+1.4)	**91.5** (+0.3)	1259 × 3 × 1	148.2
ORViT Mformer-L (Ours)	GT	IN-21K+K400	**74.9** (+6.8)	**94.2** (+3.0)	1259 × 3 × 1	148.2

(b) **Diving48**

Model	Pretrain	Frames	Top-1
I3D	K400	8	48.3
TSM	ImageNet	3	51.1
TSN	ImageNet	3	52.5
GST-50	ImageNet	8	78.9
ST-S3D	K400	8	50.6
SlowFast, R101	K400	16	77.6
TimeSformer	IN-21K	16	74.9
TimeSformer-HR	IN-21K	16	78.0
TimeSformer-L	IN-21K	96	81.0
TQN	K400	ALL	81.8
TimeSformer	IN-21K	32	80.0
TimeSformer + STIN	IN-21K	32	81.0
TimeSformer + STRG	IN-21K	32	78.1
TimeSformer + STRG + STIN	IN-21K	32	83.5
ORViT TimeSformer (Ours)	IN-21K	32	**88.0** (+8)

(c) **Epic-Kitchens100**

Method	Pretrain	A	V	N
TSN	IN-1K	33.2	60.2	46.0
TRN	IN-1K	35.3	65.9	45.4
TBN	IN-1K	36.7	66.0	47.2
TSM	IN-1K	38.3	67.9	49.0
SlowFast	K400	38.5	65.6	50.0
TimeSformer	IN-21K	32.9	55.8	50.1
ViViT-L	IN-21K+K400	44.0	66.4	56.8
Mformer	IN-21K+K400	43.1	66.7	56.5
Mformer-L	IN-21K+K400	44.1	67.1	57.6
Mformer-HR	IN-21K+K400	44.5	67.0	58.5
Mformer-HR + STIN	IN-21K+K400	44.2	67.0	57.9
Mformer-HR + STRG	IN-21K+K400	42.5	65.8	55.4
MF-HR + STRG + STIN	IN-21K+K400	44.1	66.9	57.8
ORViT Mformer-HR	IN21K+K400	**45.7** (+1.2)	**68.4** (+1.4)	**58.7** (+.2)

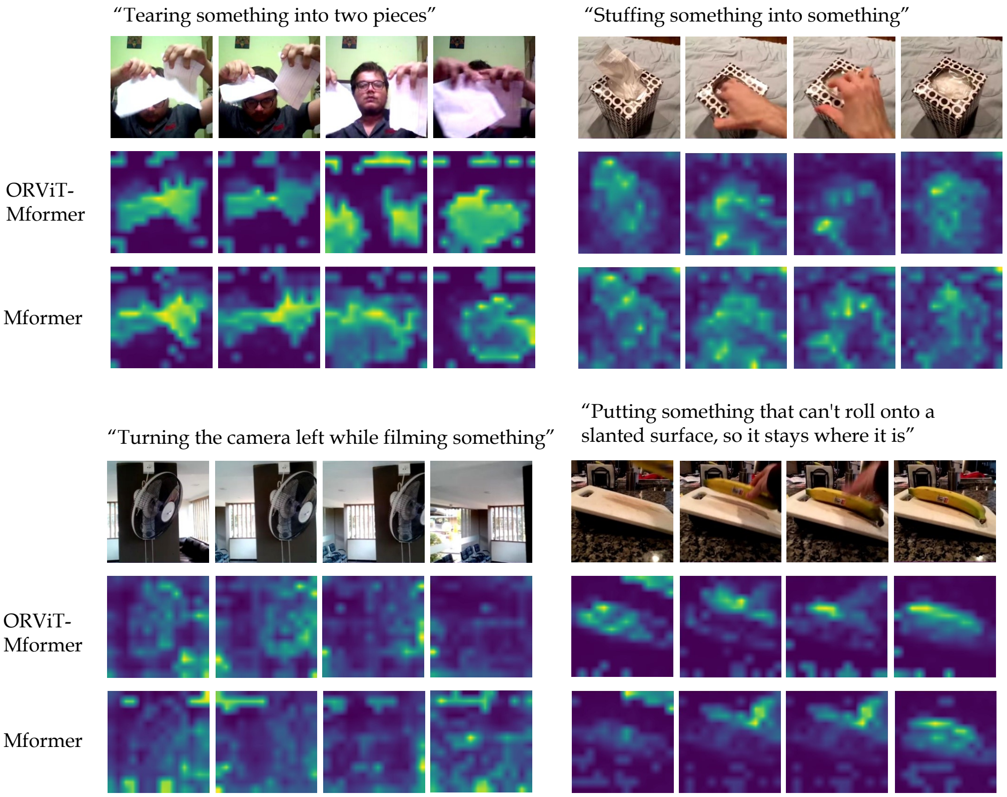

Figure 5: **Attention Maps** comparison between the *ORViT+Mformer* and the *Mformer* on videos from the SSv2 dataset. The visualization shows the attention maps corresponding to the CLS query.

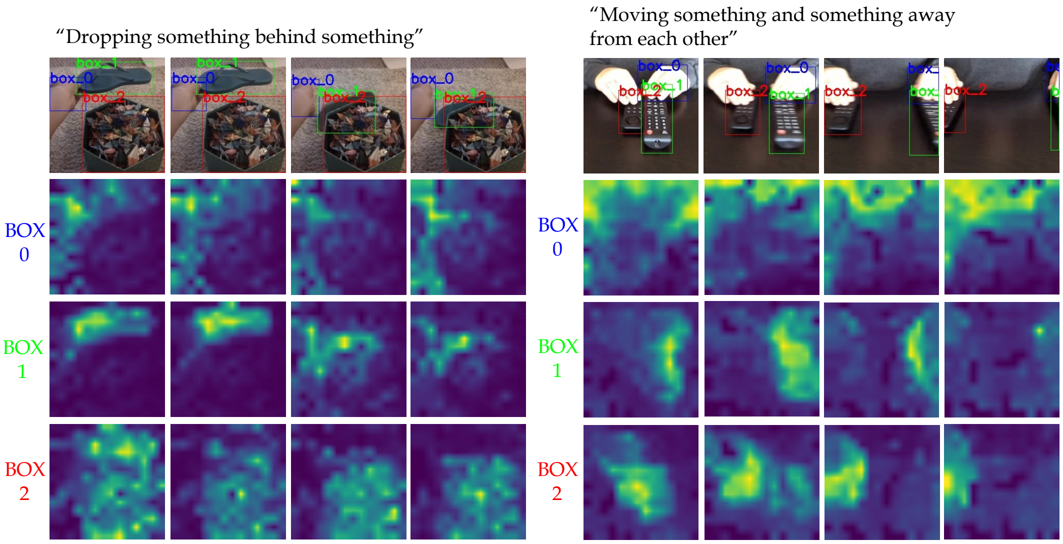

Figure 6: **Object contribution to the patch tokens.** For each object token, we plot the attention weight given by the patch tokens, normalized over the patch tokens.