# PAT: Position-Aware Transformer for Dense Multi-Label Action Detection

Faegheh Sardari[1] Armin Mustafa[1] Philip J. B. Jackson[1] Adrian Hilton[1]
[1] Centre for Vision, Speech and Signal Processing (CVSSP)
University of Surrey, UK
{f.sardari,armin.mustafa,p.jackson,a.hilton}@surrey.ac.uk

## Abstract

*We present PAT, a transformer-based network that learns complex temporal co-occurrence action dependencies in a video by exploiting multi-scale temporal features. In existing methods, the self-attention mechanism in transformers loses the temporal positional information, which is essential for robust action detection. To address this issue, we (i) embed relative positional encoding in the self-attention mechanism and (ii) exploit multi-scale temporal relationships by designing a novel non-hierarchical network, in contrast to the recent transformer-based approaches that use a hierarchical structure. We argue that joining the self-attention mechanism with multiple sub-sampling processes in the hierarchical approaches results in increased loss of positional information. We evaluate the performance of our proposed approach on two challenging dense multi-label benchmark datasets, and show that PAT improves the current state-of-the-art result by* $1.1\%$ *and* $0.6\%$ *mAP on the Charades and MultiTHUMOS datasets, respectively, thereby achieving the new state-of-the-art mAP at* $26.5\%$ *and* $44.6\%$*, respectively. We also perform extensive ablation studies to examine the impact of the different components of our proposed network.*

## 1. Introduction

Action or event detection aims to determine the boundaries of different actions/events occurring in an untrimmed video, and plays a crucial role in various important computer vision applications, such as video summarization, highlighting, and captioning. Despite the recent advances in different areas of video understanding, dense multi-label action detection is still an unsolved problem and considered as one of the most challenging video analysis tasks since the videos are untrimmed, and include several actions with different time durations that can have overlap (See Fig. 1). To carry out this task, we require to learn complex short and long term temporal relationships amongst different actions in a video which is a challenging problem [7, 15].

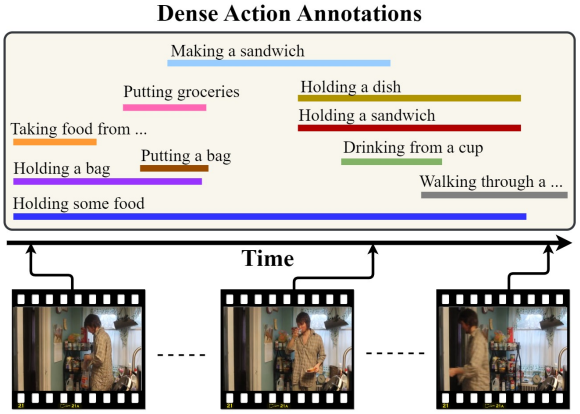



Figure 1: A sample video and its corresponding action annotations from the Charades dataset [31] where the video includes several action types with different time spans, from short to long, and in each time step, multiple actions can occur at the same time.

Most previous dense multi-label action detection approaches capture temporal dependencies through temporal convolutional networks [26, 12, 15]. However, with the success of transformer networks over the convolutional networks for modeling complex and sequential relationships [35, 9, 10, 24, 36], recently, a few methods, such as [33, 5, 7], leverage the self-attention mechanism and propose transformer-based approaches where they achieve state-of-the-art performance. Authors in [33, 5] design their network by modeling explicitly temporal cross-class relations. In [33], there are two transformer modules such that one of them investigates action relationships for each temporal moment, and another one learns temporal dependencies for each action type. However, these approaches are not computationally efficient and their complexity grows with the number of action classes. To overcome this, Dai et al. [7] design a hierarchical network that learns temporal-action dependencies from multi-scale temporal features. Their network contains several transformer layers such that the output of each layer is down-sampled and given as input into its subsequent layer. As stated in [30, 18, 11],

the self-attention mechanism in the transformer is order-invariant and loses positional information, and when the self-attention is embedded in a hierarchical structure, the issue becomes worse as using multiple down-sampling processes results in increased loss of positional information, especially in top layers. In this paper, we tackle these issues by introducing PAT, a position-aware transformer network for dense action detection. PAT consists of three main modules: fine detection, coarse detection, and classification. The fine detection module learns fine-grained action dependencies from the full temporal resolution of the video sequence for coarse detection and classification modules. The coarse detection module captures various ranges of coarse action dependencies from the fine-grained features using a non-hierarchical structure, which preserves the positional information. To further leverage the positional information, PAT incorporates a learnable relative positional encoding [29] in the transformer layers of both fine and coarse detection models. Finally, the classification module estimates the probabilities of different action classes for every timestamp in the input video using both fine and coarse-grained action dependencies. Our key contributions can be summarized as follows:

- For the first time, we introduce the idea of leveraging positional information in transformers for action detection
- We design a novel non-hierarchical transformer-based network that preserves positional information when learning multi-scale temporal action dependencies
- We evaluate the proposed method's performance on two challenging benchmark dense action detection datasets where we outperform the current state-of-the-art result by $1.1\%$ and $0.6\%$ per-frame mean average precision (mAP) on Charades and MultiTHUMOS respectively, thereby achieving the new state-of-the-art mAP at $26.5\%$ and $44.6\%$, respectively
- We perform extensive ablation studies to evaluate our network design

## 2. Related Works

Although action detection [4, 22, 20, 22, 25, 40, 34, 38, 3] has been studied significantly in computer vision, few works [27, 8, 26, 6, 15] have explored it in a dense multi-labelled setup where instances of different actions or events can overlap in different parts of a video. In this section, we review the action detection approaches by focusing on a dense-labelled setting.

To detect the boundaries of different actions, the authors in [4, 21, 23, 19] propose anchor-based methods where they first generate several proposals for each frame of video by using multi-scale anchor boxes, and then refine them to achieve the final action boundaries. However, these approaches are not usually applied for a dense multi-label scenario, as to model effectively the dense action distributions, they need a large amount of anchors [7]. To overcome this, some works, such as [27, 8, 26, 6, 15], design anchor-free approaches for dense action detection. Piergiovanni and Ryoo [27] propose a network that represents an untrimmed video into multi-activity events. They design multiple temporal Gaussian filters which are applied separately on the video frame features while a soft-attention mechanism is employed to combine the output of the filters to generate a global representation. Later in [26], they improve their work by proposing a temporal convolutional network using Gaussian filters as kernels to perform the temporal representation in a more efficient and effective way. Although they design networks to address complex multi-label action detection, the proposed models are not able to encode long-term dependencies and mostly focus on local relationships, while our proposed network is able to capture different ranges of temporal features from short to long. Kahatapitiya and Ryoo [15] propose a two-stream network to capture long term information such that one of the streams learns the most informative frame of a long video through a dynamic sub-sampling with a ratio of 4, and the other one learns the fine-grained contexts of the video from the full resolution. Although their results are promising, it cannot be adapted easily to use more temporal resolutions as it requires a dedicated Convolutional Neural Network (CNN), *i.e*. X3D [12], for each resolution, whereas in our proposed method, a different resolution can be processed easily by adding an extra branch containing a few transformer blocks in the coarse detection module.

**Transformer-based Approaches –** With the success of transformer networks in modeling complex relationships and capturing short and long term dependencies [35, 9, 10, 24, 36], some works, such as [6, 33, 7], develop transformer-based approaches for dense action detection task. Tirupattur et al. [33] design a model with two transformer branches, one branch applies self-attention across all action classes for each time step to learn the relationships amongst actions, and another branch uses self-attention across time frames to model the temporal dependencies, and the output of two branches are combined for action classification. Although this method outperforms state-of-the-art results, the method's computational complexity increases with the number of action classes. Similar to [15] that benefits different temporal resolutions, Dai et al. [7] extract multi-scale features. They design a transformer-based hierarchical structure and provide multi-resolution temporal features through several sub-sampling processes. However, as the self-attention mechanism does not preserve the temporal position information [30, 18, 11], joining it with mul-

tiple sub-sampling processes makes the network lose more positional information while preserving this information is essential for action detection. In contrast, our position-aware transformer network PAT has been designed to retain such temporal cues.

## 3. Position-Aware Transformer (PAT)

**Problem Definition –** Our aim is to detect different actions/events in a densely-labelled untrimmed video. We define the action detection problem under this setting as [15, 33, 7]. For an untrimmed video sequence with a length of $T$, each timestamp $t$ has a ground truth action label $G_t = \{g_{t,c} \in \{0,1\}\}_{c=1}^{C}$, where $C$ is the maximum number of action classes in the dataset, and the network requires to estimate action class probabilities $Y_t = \{y_{t,c} \in [0,1]\}_{c=1}^{C}$ for each timestamp.

### 3.1. Proposed Network

Our proposed method PAT is a transformer-based network designed to exploit different granularities of complex temporal dependencies for action detection. The PAT network includes a video encoder E that encodes an input video sequence into a sequence of input tokens, and three main components: fine detection module FDM, coarse detection module CDM, and classification module CLASM arranged as shown in Fig. 2. FDM processes an input sequence in its original temporal resolution to obtain a fine-grained action representation for both CDM and CLASM modules. The CDM module learns different ranges of temporal action dependencies amongst the fine-grained features through extracting and combining multi-scale temporal features. CLASM estimates class probabilities from the output of both FDM and CDM modules.

**Video Encoder (E) –** To process an input video, PAT needs to convert it into a sequence of tokens. To perform this, similar to the previous action detection approaches [33, 7, 40], we first divide the L-frame input video $V \in \mathbb{R}^{\mathrm{L}\times\mathrm{Ch}\times\mathrm{W}\times\mathrm{H}}$ into T non-overlapped segments $S = \{S_t\}_{t=1}^{T}$, where $S_t \in \mathbb{R}^{\mathrm{Z}\times\mathrm{Ch}\times\mathrm{W}\times\mathrm{H}}$, $Z = L/T$, and $Ch$, $W$, and $H$ define number of channels, width, and height of each video frame respectively. Then, the video encoder E that is a pre-trained convolutional network is employed on each segment to generate its corresponding token $I_t = E(S_t)$, where $I_t \in \mathbb{R}^{\mathrm{D}}$.

**Relative Positional Transformer (RPT) Block –** To design FDM, and CDM, we employ our proposed transformer block RPT (see Fig. 3). The RPT block comprises a transformer layer with relative positional embedding followed by a local relational LR component containing two linear layers and one 1D temporal convolutional layer as in [7] to enhance the output of the transformer layer.

As already pointed out in Section 1, the transformer self-attention mechanism loses the order of temporal information while preserving this information is essential for action detection, where we need to localise events precisely in a video sequence. To solve this issue, Vaswani et al. [35] propose to add the absolute positional embedding to the input tokens. However, in our experiments, we observed that using the absolute positional embedding decreases the method's performance significantly (see Section 4.1). This has also been observed in [7, 40]. The decrease in performance may be attributed to breaking the translation-invariant property of the method. In action detection, we expect the proposed method to be translation-invariant, *i.e.* the network learns the same representation for the same video frames in two temporally shifted videos, regardless of how much they are shifted, while the absolute encoding can break this property as it adds different positional encodings to the same frames in the shifted video inputs. To overcome this, we propose to use relative positional encoding [29] in the transformer layers of our RPT block. The relative positional encoding employs a relative pairwise distance between every two tokens and is translation-invariant. In addition, as the embedding is performed in each transformer layer and is passed into the subsequent layer, the positional information can flow to the classification module where the final estimations are provided.

We briefly formulate the transformer layer in the RPT block. In the H-head self-attention layer of RPT, for each head $h \in \{1, 2, ..., H\}$, the input sequence $X \in \mathbb{R}^{\mathrm{N}\times\mathrm{D}^{\diamond}}$ is first transferred into query $Q_h$, key $K_h$, and value $V_h$ through linear operations

$$Q_h = XW_h^q, \;\; K_h = XW_h^k, \;\; V_h = XW_h^v, \tag{1}$$

where $Q_h$, $K_h$, $V_h \in \mathbb{R}^{\mathrm{N}\times\mathrm{D_h}}$, $W_h^q$, $W_h^k$, $W_h^v \in \mathbb{R}^{\mathrm{D}^{\diamond}\times\mathrm{D_h}}$ refer the weights of linear operations, and $D_h = \frac{D^{\diamond}}{H}$. Then, the self-attention with relative positional embedding is computed for each head as

$$A_h = softmax(\frac{Q_h K_h^T + P_h^{\triangleright}}{\sqrt{D_h}})V_h, \tag{2}$$

$$P_h^{\triangleright}(n,m) = \sum_{d=1}^{D_h} Q_h(n,d)\Omega_d(n-m), \tag{3}$$

where $P_h^{\triangleright} \in \mathbb{R}^{\mathrm{N}\times\mathrm{N}}$, $n, m \in \{1, 2, ..., N\}$, and $\Omega_d$ operates as $D_h$ different embeddings for time intervals based on the queries [29]. To compute $P_h^{\triangleright}$, we use the memory-efficient method proposed by Huang et al. [13].

Finally, the self-attention of all heads are concatenated and fed into a linear layer to output sequence $O$

$$A = concat(A_1, A_2, ..., A_m), \tag{4}$$

$$O = AW^o + X, \tag{5}$$

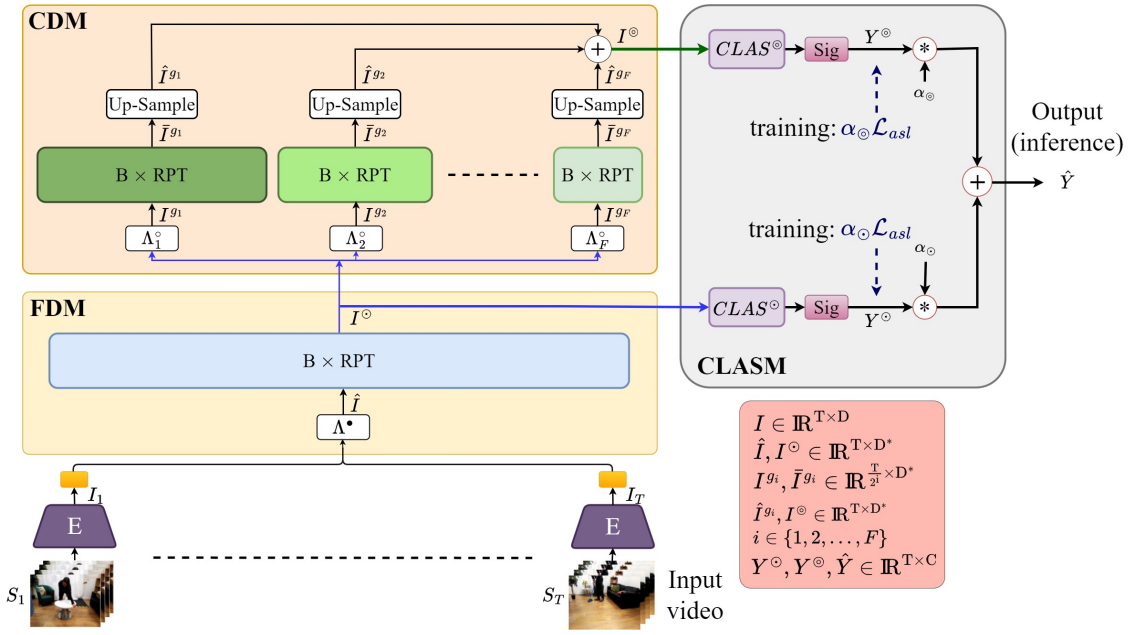



Figure 2: The overall schema of the proposed network PAT including (i) video encoder E, (ii) fine detection module FDM, (iii) Coarse detection module CDM, and (iv) classification module CLASM.

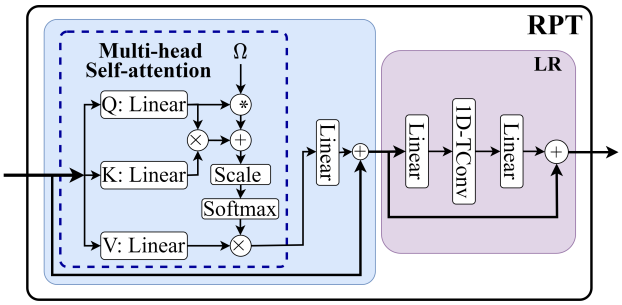



Figure 3: Architecture of the proposed RPT block. For brevity, the computation of the heads are not shown separately.

where $A \in \mathbb{R}^{\mathrm{N}\times\mathrm{D}^\diamond}$ and $W^o \in \mathbb{R}^{\mathrm{D}^\diamond\times\mathrm{D}^\diamond}$.

**Fine Detection Module (FDM) –** The FDM module aims to obtain a fine-grained temporal action dependency representation of the video from the input video sequence for the CDM and CLASM modules. FDM includes a 1D temporal convolutional layer followed by $B$ RPT blocks. The convolution layer $\Lambda^\bullet$ has a kernel size of three and a stride of one to map all the input tokens $I \in \mathbb{R}^{\mathrm{T}\times\mathrm{D}}$ into a lower dimension $D^*$, and then the RPT blocks are applied to learn the fine-grained dependencies $I^\odot$.

$$I^\odot = RPT^{FDM}_{1:B}(\Lambda^\bullet(I)), \tag{6}$$

where $I^\odot \in \mathbb{R}^{\mathrm{T}\times\mathrm{D}^*}$ and $D^* < D$.

**Coarse Detection Module (CDM) –** In the CMD module, we aim to learn a coarse temporal action dependency representation of the video. To achieve this, one solution is to extract and combine multi-scale temporal features through a hierarchical structure, such as the proposed method in [7, 40] (see Fig. 4. a). However, as we already explained in Section 1, using multiple sub-sampling processes in the hierarchical structure results in losing more positional information in the top layers of the network. Our CMD module has been designed to overcome this issue by extracting different scales of features from the same full-scale fine-grained information and through only one sub-sampling process, (see Fig. 4. b). In Section 4.1, we show that our novel non-hierarchical design to extract multi-scale features outperforms significantly a hierarchical structure.

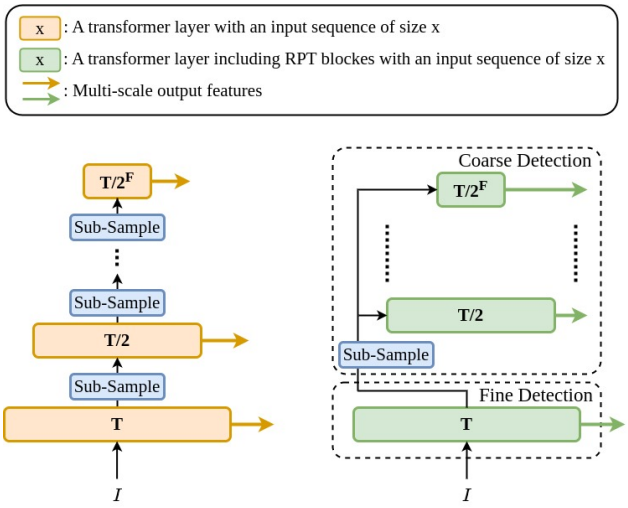
The CMD module has $F$ granularity branches such that each branch learns a different scale of temporal features. In the $i^{th}$ branch, first a 1D temporal convolutional layer $\Lambda^\circ_i$ with a kernel size of three and a stride of $2^i$ is applied on the fine-grained inputs received from the preceding module FCM as

$$I^{g_i} = \Lambda^\circ_i(I^\odot), \tag{7}$$

where $I^{g_i} \in \mathbb{R}^{\mathrm{T}^\mathrm{i}\times\mathrm{D}^*}$, $i \in \{1, 2, ..., F\}$, and $T^i = \frac{T}{2^i}$. Then, the down-sampled features are given into $B$ RPT transformer blocks to exploit the temporal dependencies amongst them

$$\bar{I}^{g_i} = RPT^{CDM_i}_{1:B}(I^{g_i}), \tag{8}$$

where $\bar{I}^{g_i} \in \mathbb{R}^{\mathrm{T}^\mathrm{i}\times\mathrm{D}^*}$. Note, to extract all the scales of features, the striding process (sub-sampling) is used only




(a) Hierarchical structure (b) Our proposed structure

Figure 4: The proposed hierarchical structure in [7, 40] vs. our proposed non-hierarchical design in fine and coarse detection modules to extract multi-scale features for action detection.

once and as the relative positional information has been already embedded in the fine-grained information, after the striding process, the sub-sample features keep the temporal positional cues.

In the CLASM module, action class probabilities are estimated for each input token generated by the video encoder E. Therefore, the CMD module requires obtaining a coarse dependency representation at the original temporal length. To do this, we up-sample and combine different scales of coarse features to provide a final coarse representation $I^{\circledcirc}$ as

$$\hat{I}^{g_i} = UpSample(\bar{I}^{g_i}), \tag{9}$$

$$I^{\circledcirc} = \sum_{i=1}^{F} \hat{I}^{g_i}, \tag{10}$$

where $\hat{I}^{g_i}, I^{\circledcirc} \in \mathbb{R}^{\mathrm{T}\times\mathrm{D}^*}$ and linear interpolation is employed for up-sampling.

**Classification Module (CLASM) –** This module obtains the action class probabilities for action detection from the fine and coarse contexts. To this, two convolution blocks $CLAS^{\odot}$ and $CLAS^{\circledcirc}$ that include two 1D convolution filters with kernel size one and stride one are applied on the fine and coarse features separately to predict $C$ action class probabilities for each temporal moment

$$Y^{\phi} = Sig(CLAS^{\phi}(I^{\phi})), \tag{11}$$

where $Y^{\phi} \in \mathbb{R}^{\mathrm{T}\times\mathrm{C}}$, $\phi \in \{\odot, \circledcirc\}$, and $Sig$ refers to sigmoid activation function. Then, at inference, the final estimation is computed by combining them as

$$\hat{Y} = \sum_{\phi} \alpha_{\phi} Y^{\phi}, \tag{12}$$

where $\hat{Y} \in \mathbb{R}^{\mathrm{T}\times\mathrm{C}}$ and $\alpha_{\odot} + \alpha_{\circledcirc} = 1$.

### 3.2. Network Optimization

To optimize action detection models, binary cross entropy (BCE) is usually used as in [33, 7, 40, 34]. However, in the multi-label setting, the number of positive labels may become more than the number of negative ones. This unbalanced number of positive and negative labels can result in poor performance in the action detection task if we employ BCE for training, since it does not have any control on the contribution of positive and negative samples. To overcome this, we propose to adapt Asymmetric loss $\mathcal{L}_{asl}$ [28] for multi-label action detection. Therfore, the total loss $\mathcal{L}_{total}$ is computed as

$$\mathcal{L}_{total} = \frac{1}{T}\sum_{\phi}\sum_{t=1}^{T}\sum_{c=1}^{C} \alpha_{\phi}\mathcal{L}_{asl}(g_{t,c}, y_{t,c}^{\phi}), \tag{13}$$

$$\mathcal{L}_{asl}(g_{t,c}, y_{t,c}^{\phi}) = -g_{t,c}\mathcal{L}_{+} - (1-g_{t,c})\mathcal{L}_{-}, \tag{14}$$

$$\mathcal{L}_{+} = (1-y_{t,c}^{\phi})^{\gamma_{+}} log(y_{t,c}^{\phi}), \tag{15}$$

$$\mathcal{L}_{-} = (\acute{y}_{t,c}^{\phi})^{\gamma_{-}} log(1-\acute{y}_{t,c}), \tag{16}$$

$$\acute{y}_{t,c}^{\phi} = max(y_{t,c}^{\phi} - \delta, 0), \tag{17}$$

where $g_{t,c}$ indicates the ground truth label of action class $c$ in temporal step $t$, and $y_{t,c}^{\phi}$ is its corresponding class probability estimated by Eq. 11. $\gamma_{+}$ and $\gamma_{-}$ are focusing parameters for positive and negative labels respectively and if we choose $\gamma_{+} < \gamma_{-}$, we are able to increase the contribution of positive samples. Furthermore, Eq. 17 applies another asymmetric mechanism by discarding the very easy negative samples through setting the threshold parameter $\delta$. In Section 4.1, we show that optimizing the proposed network through Asymmetric loss instead of BCE improves the method's performance.

## 4. Experimental Results

**Datasets –** There are several benchmark datasets for action detection, but only a few of them provide dense multi-label annotations. For instance, videos in ActivityNet [1] have only one action type per timestamp. We present the results of PAT on two challenging dense multi-label benchmark datasets, Charades [31] and MultiTHUMOS [39].

Charades [31] is a large dataset including $9,848$ videos of daily activities of 267 persons. It contains $66,500$ temporal interval annotations for 157 action classes while there is a high overlap amongst the action instances of different

action categories. To evaluate our method on Charades, we follow previous methods [15, 33, 7] and use the same training and testing set as in [31].

MultiTHUMOS contains the same set of 413 videos as in THUMOS'14 dataset [14]. However, MultiTHUMOS is more challenging than THUMOS'14 since (i) the annotations have been extended from 20 action classes to 65, and (ii) in contrast to sparse-lable frame-level annotations in THUMOS'14, MultiTHUMOS has dense multi-label action annotations. To obtain the results on this dataset, we use the same standard training and testing splits applied by previous methods [33, 7]. Following state-of-the-art methods [27, 26, 15, 33, 7], we evaluate our method on these datasets by standard per-frame mAP metric.

**Implementation Details –** Similar to the proposed method in [7], during both training and inference, PAT uses a fixed number of $T = 256$ input tokens. For training, we randomly sample a clip containing $T$ consecutive tokens from a video sequence. At inference, we follow previous work [33, 15] and make the predictions for a full video sequence. Each input token is provided by applying the video encoder E on an 8-frame segment to extract a feature vector with dimension $D = 1024$. The video encoder E is implemented by using a pre-trained I3D [2][1] while its fully connected layers are replaced with a global average pooling layer and its parameters are frozen. In the convolutional layer of FDM, the input features are mapped into $D^* = 512$ dimensional feature vectors. Note, the feature dimension $D^* = 512$ is fixed for the rest of the network. FDM and each granularity branch of CDM have $B = 3$ RPT blocks with $H = 8$ multi-head attention heads, and the number of granularity branches in CMD is set to $F = 3$ as we found that with these parameters, PAT obtains the best performance. The contributing factors for fine-grained ($\alpha_{\odot}$) and coarse-grained ($\alpha_{\circledcirc}$) features in the CLASM module are set empirically to $\{\alpha_{\odot} = 0.1, \alpha_{\circledcirc} = 0.9\}$ and $\{\alpha_{\odot} = 0.7, \alpha_{\circledcirc} = 0.3\}$ for Charades and MultiTHUMOS respectively. In Asymmetric loss, we use factors of $\gamma_+ = 1$ and $\gamma_- = 3$ for the impact of positive and negative samples respectively, and threshold parameter $\delta = 0.1$, which are determined through trial and error.

Our experiments were performed under Pytorch on an NVIDIA GeForce RTX 3090 GPU, and we trained our model using the Adam optimiser [17] with an initial learning rate of 0.0001 and batch size 3 for 25 and 300 epochs for Charades and MultiTHUMOS datasets respectively. The learning rate was decreased by a factor of 10 every 7 and 130 epochs for Charades and MultiTHUMOS respectively. Note, using different training settings for Charades and MultiTHUMOS is due to their different size.

[1] Video encoder E is pre-trained on Kinetic-400 [16] and training set of Charades for MultiTHUMOS and Charades respectively.

### 4.1. Ablation Studies

In this section, we examine our design decisions for the proposed network and learning paradigm.

**Effect of FDM and CDM Modules –** Here, we aim to evaluate the impact of the fine and coarse detection modules (FDM and CDM) in the final results of PAT. Table 1 shows per-frame mAP on the Charades and MultiTHUMOS datasets as we remove each or both of FDM and CDM modules. To obtain the results of the network when both modules are dropped, we use directly the sequence of input tokens generated by the video encoder (I3D) for action detection. Table 1 shows that using only input tokens generated by I3D network is not enough for effective action detection and employing fine and coarse-grained temporal features obtained by FDM and CDM improves the performance by $9.7\%$ and $7.9\%$ per-frame mAP on Charades and MultiTHUMOS respectively. It also shows that both FDM and CDM modules have an important contribution to the final results as by removing FDM and CDM, our results deteriorate by $2.4\%$ and $3.4\%$ per-frame mAP on average on both datasets respectively. Table 1 also shows that for different datasets that have different action types, the contribution of fine and coarse features might be different which is the reason we use the contribution factors $\{\alpha_{\odot}, \alpha_{\circledcirc}\}$ to combine the prediction results of FDM and CDM in the CLASM module at the inference.

| Module | mAP(%) | |
|---|---|---|
| | Charades | MultiTHUMOS |
| CLASM | 16.8 | 36.7 |
| FDM, CLASM | 23.8 | 40.5 |
| CDM, CLASM | 26.2 | 40.1 |
| FDM, CDM, CLASM | **26.5** | **44.6** |

Table 1: Ablation studies on FDM and CDM modules of PAT on the Charades and MultiTHUMOS dataset using RGB videos in terms of per-frame mAP metric.

**Effect of Structure Design to Extract Multi-Scale Features –** In this section, we examine the design of PAT with two other variants to capture fine-grained and coarse-grained features. In the first variant PAT-$v_1$, the CDM module uses the hierarchical structure to extract the multi-scale features while the rest of its architecture is the same as PAT. In the second variant PAT-$v_2$, the CDM module has a non-hierarchical structure, the same as PAT, but the FDM module and all granularity branches in CDM learn their features from input tokens.

Table 2 shows that when CMD applies a hierarchical structure to learn the coarse-grained features, *i.e.* PAT-$v_1$, the method's performance drops $1.4\%$ and $0.6\%$ on Charades and MultiTHUMOS respectively. This proves

| Design | mAP(%) | |
|---|---|---|
| | Charades | MultiTHUMOS |
| PAT-$v_1$ (Hierarchical) | 25.1 | 44.0 |
| PAT-$v_2$ | 26.1 | 44.2 |
| PAT | **26.5** | **44.6** |

Table 2: Ablation studies on structure design of the proposed method on the Charades and MultiTHUMOS datasets using RGB videos in terms of per-frame mAP metric.

the contribution of our novel non-hierarchical transformer-based design which preserves positional information when exploiting the multi-scale features. Furthermore, in case we apply a non-hierarchical CMD, *i.e*. PAT and PAT-$v_2$, if the CMD module extracts the multi-scale features from the fine-grained context instead of the input tokens as in PAT, we achieve the best performance at $26.5\%$ and $44.6\%$ per-frame mAP on Charades and MultiTHUMOS respectively.

**Impact of Relative Positional Encoding –** Table 3 shows the performance of PAT when different positional encodings are applied. It can be observed that employing the relative positional encoding [29, 13] embedded in the RPT block improves the method's performance by $0.3\%$ per-frame mAP on both datasets, while adding absolute positional encoding [35] into the input tokens deteriorates the method's performance significantly.

| Positional Encoding | mAP(%) | |
|---|---|---|
| | Charades | MultiTHUMOS |
| No encoding | 26.2 | 44.3 |
| Absolute | 25.3 | 43.5 |
| Relative | **26.5** | **44.6** |

Table 3: Ablation studies on positional encoding used in PAT on the Charades and MultiTHUMOS dataset using RGB videos in terms of per-frame mAP metric.

**Impact of Loss Function –** Here, we examine the effect of BCE and Asymmetric [28] losses for training. As shown in Table 4, applying the Asymmetric loss [28] to optimize PAT improves the performance by $0.5\%$ and $0.2\%$ per-frame mAP on Charades and MultiTHUMOS respectively.

| Loss | mAP(%) | |
|---|---|---|
| | Charades | MultiTHUMOS |
| BCE | 26.0 | 44.4 |
| Asymmetric [28] | **26.5** | **44.6** |

Table 4: Ablation studies on the loss function applied for training PAT on the Charades and MultiTHUMOS datasets using RGB videos in terms of per-frame mAP metric.

**Discussion and Analysis –** The ablation studies show that leveraging positional information in the transformer layers has an important contribution in the final results of the network where extracting the multi-scale temporal features through our proposed non-hierarchical design in CMD outperforms a hierarchical structure by $1.0\%$ mAP on average on both datasets (PAT vs PAT-v1), and embedding the relative position encoding in the RPT block improves the performance by $0.3\%$ mAP on both datasets. Our further ablations also reveal the effect of the Asymmetric loss in optimizing of PAT where it increases the performance by $0.3\%$ mAP on average on both datasets.

### 4.2. State-of-the-Art Comparison

In this section, we compare the performance of the proposed method with the state-of-the-art action detection approaches including both transformer-based methods and the methods that do not use self-attention. Both quantitative and qualitative results are obtained for this section.

Table 5 provides comparative results on the benchmark datasets Charades and MultiTHUMOS based on the standard per-frame mAP metric. Table 5 shows that our proposed method outperforms the current state-of-the-art result by $1.1\%$ and $0.6\%$ on Charades and MultiTHUMOS respectively and achieves a new state-of-the-art per-frame mAP results at $26.5\%$ and $44.6\%$ on Charades and MultiTHUMOS respectively.

We also evaluate the performance of our proposed method by action-conditional metrics including Action-Conditional Precision $P_{AC}$, Action-Conditional Recall $R_{AC}$, Action-Conditional F1-Score $F1_{AC}$, and Action-Conditional Mean Average Precision $mAP_{AC}$, as introduced in [33]. The aim of these metrics is to measure the ability of the network to learn both co-occurrence and temporal dependencies of different action classes. The metrics are measured throughout a temporal window with a size of $\tau$. As shown by the results on Charades in Table 6, the proposed method PAT achieves state-of-the-art results on all action-conditional metrics, specifically, it improves the state-of-the-art results significantly on $R_{AC}$ and $F1_{AC}$ by $10.6\%$ and $7.7\%$, $10.8\%$ and $7.5\%$, and $10.8\%$ and $7.3\%$ where $\tau$ is 0, 20, and 40 respectively.

Fig. 5 displays qualitative results of PAT on a test video sample of Charades and compares them with the outputs of MS-TCT [7]. Amongst the state-of-the-art methods, we applied MS-TCT [7] and MLAD [33] on the video sample, since their code is available, useable and compatible with our hardware. However, as the MLAD could not predict any of the actions, we reported only the results of MS-TCT. The results in Fig. 5 show that our proposed method's action predictions have a better overlap with the ground-truth labels, and our method detected more action instances in the video than MS-TCT, *i.e*. PAT predicted all action types ex-

| Method | | GFLOPs | Backbone | mAP(%) | |
|---|---|---|---|---|---|
| | | | | Charades | MultiTHUMOS |
| R-C3D [37] | ICCV 2017 | - | C3D | 12.7 | - |
| SuperEvent [27] | CVPR 2018 | 0.8 | I3D | 18.6 | 36.4 |
| TGM [26] | ICML 2019 | 1.2 | I3D | 20.6 | 37.2 |
| PDAN [6]✓∗ | WACV 2021 | 3.2 | I3D | 23.7 | 40.2 |
| CoarseFine [15] | CVPR 2021 | - | X3D | 25.1 | - |
| MLAD [33]✓ | CVPR 2021 | 44.8 | I3D | 18.4 | 42.2 |
| CTRN [5]✓ | BMVC 2021 | - | I3D | 25.3 | 44.0 |
| PointTAD [32] | NeurIPS 2022 | - | I3D | 21.0 | 39.8 |
| MS-TCT [7]✓ | CVPR 2022 | 6.6 | I3D | 25.4 | 43.1 |
| PAT✓ | | 8.5 | I3D | **26.5** | **44.6** |

Table 5: Action detection results on Charades and MultiTHUMOS datasets using RGB videos in terms of per-frame mAP. The ✓ symbol highlights the transformer-based approaches, and ∗ indicates the results are taken from [7].

| Method | $\tau = 0$ | | | | $\tau = 20$ | | | | $\tau = 40$ | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $P_{AC}$ | $R_{AC}$ | $F1_{AC}$ | $mAP_{AC}$ | $P_{AC}$ | $R_{AC}$ | $F1_{AC}$ | $mAP_{AC}$ | $P_{AC}$ | $R_{AC}$ | $F1_{AC}$ | $mAP_{AC}$ |
| I3D[2]∗ | 14.3 | 1.3 | 2.1 | 15.2 | 12.7 | 1.9 | 2.9 | 21.4 | 14.9 | 2.0 | 3.1 | 20.3 |
| CF [33]∗ | 10.3 | 1.0 | 1.6 | 15.8 | 9.0 | 1.5 | 2.2 | 22.2 | 10.7 | 1.6 | 2.4 | 21.0 |
| MLAD [33]✓ | 19.3 | 7.2 | 8.9 | 28.9 | 18.9 | 8.9 | 10.5 | 35.7 | 19.6 | 9.0 | 10.8 | 34.8 |
| MS-TCT [7]✓ | 26.3 | 15.5 | 19.5 | 30.7 | 27.6 | 18.4 | 22.1 | 37.6 | 27.9 | 18.3 | 22.1 | 36.4 |
| PAT✓ | **28.3** | **26.1** | **27.2** | **32.0** | **30.0** | **29.2** | **29.6** | **37.8** | **30.0** | **29.1** | **29.4** | **36.7** |

Table 6: Action detection results on Charades dataset based on the action-conditional metrics [33], $P_{AC}$, $R_{AC}$, $F1_{AC}$, and $mAP_{AC}$. $\tau$ refers the temporal window size. The same as [7, 33], both RGB and optical flow are used for obtaining the results. The ✓ symbol highlights the transformer-based approaches, and ∗ indicates the results are taken from [33].

cept "*Taking a bag*" while MS-TCT could not detect "*Taking a picture*", "*Taking a bag*", and "*Walking*".

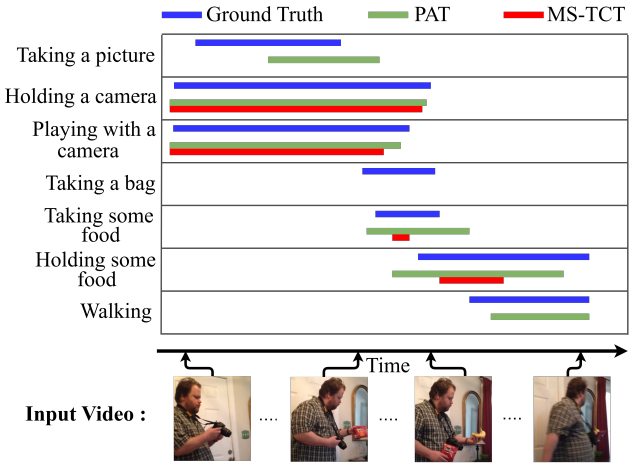



Figure 5: Visualization of action predictions by our proposed method PAT and MS-TCT [7] on a test video sample of Charades including 7 different action types.

## 5. Conclusion

In this work, we introduced a novel transformer-based network PAT that exploits different ranges of temporal dependencies for action detection. The proposed method has been designed to benefit from preserving temporal positional information in learning multi-granularity features by (i) embedding the relative positional encoding in its transformer layers and (ii) a non-hierarchical design. We evaluated PAT on two densely-labelled challenging benchmark action detection datasets, on which we achieved new state-of-the-art results, and our ablation studies demonstrated the effectiveness of different components of our proposed network. For future work, we will investigate adapting our network to learn spatial and temporal dependencies from raw pixels and also use audio information to improve the performance of action detection.

## Acknowledgement

This research is supported by UKRI EPSRC Platform Grant EP/P022529/1, and EPSRC BBC Prosperity Partnership AI4ME: Future Personalised Object-Based Media Experiences Delivered at Scale Anywhere EP/V038087/1.

## References


[1] Fabian Caba Heilbron, Victor Escorcia, Bernard Ghanem, and Juan Carlos Niebles. ActivityNet: A large-Scale Video Benchmark for Human Activity Understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 961–970, 2015. 5

[2] 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*, pages 6299–6308, 2017. 6, 8

[3] Shuning Chang, Pichao Wang, Fan Wang, Hao Li, and Jiashi Feng. Augmented Transformer with Adaptive Graph for Temporal Action Proposal Generation. *arXiv preprint arXiv:2103.16024*, 2021. 2

[4] 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*, pages 1130–1139, 2018. 2

[5] Rui Dai, Srijan Das, and Francois Bremond. Ctrn: Class-temporal relational network for action detection. *British Machine Vision Conference*, 2021. 1, 8

[6] Rui Dai, Srijan Das, Luca Minciullo, Lorenzo Garattoni, Gianpiero Francesca, and Francois Bremond. PDAN: Pyramid Dilated Attention Network for Action Detection. In *Proceedings of the IEEE Winter Conference on Applications of Computer Vision*, pages 2970–2979, 2021. 2, 8

[7] Rui Dai, Srijan Das, Kumara Kahatapitiya, Michael S Ryoo, and Francois Bremond. MS-TCT: Multi-Scale Temporal ConvTransformer for Action Detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 20041–20051, 2022. 1, 2, 3, 4, 5, 6, 7, 8

[8] Xiyang Dai, Bharat Singh, Joe Yue-Hei Ng, and Larry Davis. TAN: Temporal Aggregation Network for Dense Multi-Label Action Recognition. In *2019 IEEE Winter Conference on Applications of Computer Vision*, pages 151–160. IEEE, 2019. 2

[9] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding. *arXiv preprint arXiv:1810.04805*, 2018. 1, 2

[10] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An Image Is Worth 16x16 Words: Transformers for Image Recognition at Scale. *International Conference on Learning Representations*, 2020. 1, 2

[11] Philipp Dufter, Martin Schmitt, and Hinrich Schütze. Position Information in Transformers: An Overview. *Computational Linguistics*, 48(3):733–763, 2022. 1, 2

[12] Christoph Feichtenhofer. X3D: Expanding Architectures for Efficient Video Recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 203–213, 2020. 1, 2

[13] Cheng-Zhi Anna Huang, Ashish Vaswani, Jakob Uszkoreit, Noam Shazeer, Ian Simon, Curtis Hawthorne, Andrew M Dai, Matthew D Hoffman, Monica Dinculescu, and Douglas Eck. Music Transformer. *International Conference on Learning Representations*, 2019. 3, 7

[14] Yu-Gang Jiang, Jingen Liu, A Roshan Zamir, George Toderici, Ivan Laptev, Mubarak Shah, and Rahul Sukthankar. THUMOS Challenge: Action Recognition with a Large Number of Classes, 2014. 6

[15] Kumara Kahatapitiya and Michael S Ryoo. Coarse-Fine Networks for Temporal Activity Detection in Videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8385–8394, 2021. 1, 2, 3, 6, 8

[16] 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. 6

[17] Diederik P Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. *International Conference on Learning Representations*, 2015. 6

[18] Yang Li, Si Si, Gang Li, Cho-Jui Hsieh, and Samy Bengio. Learnable Fourier Features for Multi-Dimensional Spatial Positional Encoding. *Advances in Neural Information Processing Systems*, 34:15816–15829, 2021. 1, 2

[19] Zhihui Li and Lina Yao. Three Birds with One Stone: Multi-Task Temporal Action Detection via Recycling Temporal Annotations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4751–4760, 2021. 2

[20] Chuming Lin, Chengming Xu, Donghao Luo, Yabiao Wang, Ying Tai, Chengjie Wang, Jilin Li, Feiyue Huang, and Yanwei Fu. Learning Salient Boundary Feature for Anchor-Free Temporal Action Localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3320–3329, 2021. 2

[21] Tianwei Lin, Xu Zhao, Haisheng Su, Chongjing Wang, and Ming Yang. BSN: Boundary Sensitive Network for Temporal Action Proposal Generation. In *Proceedings of the European Conference on Computer Vision*, pages 3–19, 2018. 2

[22] Tianwei Lin, Xiao Liu, Xin Li, Errui Ding, and Shilei Wen. BMN: Boundary-Matching Network for Temporal Action Proposal Generation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3889–3898, 2019. 2

[23] Yuan Liu, Lin Ma, Yifeng Zhang, Wei Liu, and Shih-Fu Chang. Multi-Granularity Generator for Temporal Action Proposal. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3604–3613, 2019. 2

[24] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin Transformer: Hierarchical Vision Transformer Using Shifted Windows. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 10012–10022, 2021. 1, 2

[25] Fan Ma, Linchao Zhu, Yi Yang, Shengxin Zha, Gourab Kundu, Matt Feiszli, and Zheng Shou. SF-Net: Single-Frame Supervision for Temporal Action Localization. In *Proceedings of the European Conference on Computer Vision*, pages 420–437, 2020. 2

[26] AJ Piergiovanni and Michael Ryoo. Temporal Gaussian Mixture Layer for Videos. In *International Conference on Machine learning*, pages 5152–5161. PMLR, 2019. 1, 2, 6, 8

[27] AJ Piergiovanni and Michael S Ryoo. Learning Latent Super-Events to Detect Multiple Activities in Videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5304–5313, 2018. 2, 6, 8

[28] Tal Ridnik, Emanuel Ben-Baruch, Nadav Zamir, Asaf Noy, Itamar Friedman, Matan Protter, and Lihi Zelnik-Manor. Asymmetric Loss For Multi-Label Classification. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 82–91. IEEE Computer Society, 2021. 5, 7

[29] Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. Self-Attention with Relative Position Representations. *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2, 2018. 2, 3, 7

[30] Tao Shen, Tianyi Zhou, Guodong Long, Jing Jiang, Shirui Pan, and Chengqi Zhang. DiSAN: Directional Self-Attention Network for RNN/CNN-Free Language Understanding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018. 1, 2

[31] Gunnar A Sigurdsson, Gül Varol, Xiaolong Wang, Ali Farhadi, Ivan Laptev, and Abhinav Gupta. Hollywood in Homes: Crowdsourcing Data Collection for Activity Understanding. In *Proceedings of the European Conference on Computer Vision*, pages 510–526, 2016. 1, 5, 6

[32] Jing Tan, Xiaotong Zhao, Xintian Shi, Bin Kang, and Limin Wang. PointTAD: Multi-Label Temporal Action Detection with Learnable Query Points. In *Advances in Neural Information Processing Systems*, 2022. 8

[33] Praveen Tirupattur, Kevin Duarte, Yogesh S Rawat, and Mubarak Shah. Modeling Multi-Label Action Dependencies for Temporal Action Localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1460–1470, 2021. 1, 2, 3, 5, 6, 7, 8

[34] Elahe Vahdani and Yingli Tian. Deep Learning-based Action Detection in Untrimmed Videos: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022. 2, 5

[35] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention Is All You Need. *Advances in Neural Information Processing Systems*, 30, 2017. 1, 2, 3, 7

[36] Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pyramid Vision Transformer: A Versatile Backbone for Dense Prediction without Convolutions. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 568–578, 2021. 1, 2

[37] Huijuan Xu, Abir Das, and Kate Saenko. R-C3D: Region Convolutional 3D Network for Temporal Activity Detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5783–5792, 2017. 8

[38] Mengmeng Xu, Chen Zhao, David S Rojas, Ali Thabet, and Bernard Ghanem. G-TAD: Sub-Graph Localization for Temporal Action Detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 10156–10165, 2020. 2

[39] Serena Yeung, Olga Russakovsky, Ning Jin, Mykhaylo Andriluka, Greg Mori, and Li Fei-Fei. Every Moment Counts: Dense Detailed Labeling of Actions in Complex Videos. *International Journal of Computer Vision*, 126(2):375–389, 2018. 5

[40] Chenlin Zhang, Jianxin Wu, and Yin Li. ActionFormer: Localizing Moments of Actions with Transformers. *Proceedings of the European Conference on Computer Vision*, 2022. 2, 3, 4, 5