# Few-Shot Transformation of Common Actions into Time and Space

Pengwan Yang, Pascal Mettes, Cees G. M. Snoek
University of Amsterdam

## Abstract

*This paper introduces the task of few-shot common action localization in time and space. Given a few trimmed support videos containing the same but unknown action, we strive for spatio-temporal localization of that action in a long untrimmed query video. We do not require any class labels, interval bounds, or bounding boxes. To address this challenging task, we introduce a novel few-shot transformer architecture with a dedicated encoder-decoder structure optimized for joint commonality learning and localization prediction, without the need for proposals. Experiments on re-organizations of the AVA and UCF101-24 datasets show the effectiveness of our approach for few-shot common action localization, even when the support videos are noisy. Although we are not specifically designed for common localization in time only, we also compare favorably against the few-shot and one-shot state-of-the-art in this setting. Lastly, we demonstrate that the few-shot transformer is easily extended to common action localization per pixel.*

## 1. Introduction

The goal of this paper is to localize an action in video time and space, without the need for class labels, interval bounds or box annotations. Class-agnostic action proposals, for either temporal, *e.g.* [13, 25, 32] or spatio-temporal, *e.g.*, [18, 48, 50] action localization, have the same goal. However, to be effective they need to generate many proposals and a secondary *supervised* step to find the best fitting one. To avoid the need for extensive supervision, Feng *et al*. [7] introduced *one-shot* localization of actions in time and space. They rely on proposals as well, but rather than using class supervision, a matching model between a trimmed support video and a long untrimmed video determines the best proposal. In similar spirit, Yang *et al*. [45] introduced common action localization in *time*. Given a few trimmed support videos containing the same (unknown) action, they are able to localize with the aid of proposals an action in a long untrimmed video. Their setup also avoids the need for temporal and class annotations. In this work, we extend upon both [7] and [45] and propose the new task of *few-shot*
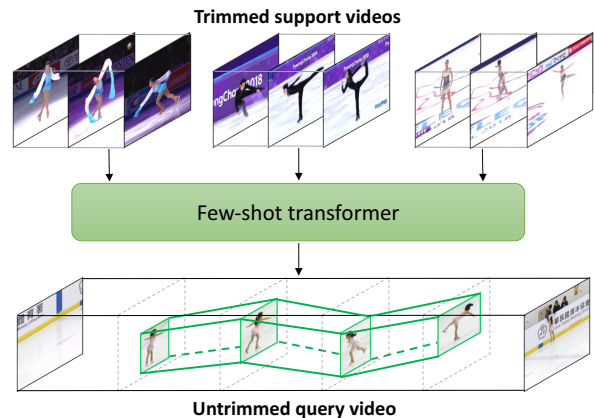


Figure 1: **Few-shot common action localization in time and space.** Given a few trimmed support videos sharing a common action, our proposed few-shot transformer is able to localize the spatio-temporal tubelet of the common action in a long untrimmed query video, without requiring the action class label, or any temporal or spatial annotations.

common action localization in *time and space*. Our approach does not require any box annotations or class labels to obtain the spatio-temporal localization, and neither do we need proposals as in [7, 45]. All we require are a handful of trimmed videos showing a common unnamed action, see Figure 1.

Our approach to few-shot common action localization is inspired by the success of transformers [1] in object detection as demonstrated by Carion *et al*. [3]. Their approach eliminates the need for proposals and the accompanying hand-crafted components, while maintaining competitive performance. Such a proposal-free method avoids the needle-in-the-haystack problem with proposals in object detection, a problem which is even more severe in spatio-temporal action localization. Further, they exploit the versatile and powerful relation modeling capability of transformers. This is naturally suitable for common action localization as well, where the core challenge is to model the commonality between the few support videos and the single query video. To that end, we propose a transformer with an encoder-decoder structure that is adapted for our task of few-shot common action localization in time and space.

We make three contributions. First, we introduce the task of few-shot common action localization in time and space. We localize the spatio-temporal tube encapsulating an unknown action, based only on the commonality between a long untrimmed query video and a few trimmed support videos containing the same unknown action. Second, we propose a few-shot transformer with a dedicated encoder-decoder structure optimized for joint commonality learning and localization prediction, without the need for proposals. As a third contribution, we reorganize the videos in AVA [11] and UCF101-24 [36] to allow for evaluation of few-shot common action localization in time and space. Our experiments show the effectiveness of our approach, even when support videos are noisy. Moreover, we demonstrate compatibility to few-shot and one-shot temporal action localization, outperforming the respective state-of-the-art. Last but not least, we show that the few-shot transformer easily extends to common action localization per pixel. Code: `https://github.com/PengWan-Yang/few-shot-transformer`.

## 2. Related work

**Few- and zero-shot action localization.** Yang *et al.* [44] introduce few-shot action-class localization in time, where a few (or at least one) positive labeled and several negative labeled videos steer the localization via an end-to-end meta-learning strategy. The strategy relies on sliding windows to swipe over the untrimmed query video to generate fixed boundary proposals. Xu *et al.* [43] also temporally localize an action from a few positive labeled and several negative labeled videos. They adopt a region proposal network [31] to produce proposals with flexible boundaries. Zhang *et al.* [49] perform few-shot temporal action localization, where video-level annotations are needed. They construct a multi-scale feature pyramid to directly produce temporal features at variable scales. For spatio-temporal action localization, a number of works have investigated a zero-shot perspective by linking actions to relevant objects [19, 21, 28, 29], or by leveraging trimmed videos used for action classification [17]. Soomro *et al.* [35] tackle action localization in an unsupervised setting, where no annotations are provided at all. While zero-shot and unsupervised action localization show promise, current approaches are not competitive with few-shot alternatives, hence we focus on the few-shot setting. Rather then relying on a few positive and many negative action class labels, like [43, 44], our action localization approach does not require any predefined positive nor negative action class labels, neither does it need any temporal or spatial bounds. All we require is that the few trimmed support videos have the same action in *common*. Moreover, we attempt to localize the common action both in time and in space.

**Localization by commonality.** Compared with few-shot localization, common localization further frees the need for labels. It localizes according to the commonality between the query and support inputs. Kang *et al.* [22] and Hu *et al.* [14] detect the common object in an image by a few examples containing the same object, without the need to know their class name. They both adopt a one-stage detector architecture with a feature reweighting module. Also for common object detection, Fan *et al.* [6] rely on the support-guided region proposal network in a two-stage approach to produce support-related proposals and match each proposal with support images. Video relocalization by Feng *et al.* [8] introduces temporal action localization in an untrimmed query video from a *single* unlabeled support video, along with a one-stage approach. Yang *et al.* [45] propose few-shot temporal localization of the common action. While their two-stage architecture is more general and effective than [8], it depends on proposals and several hand-crafted components, such as anchors in the proposal generation and non-maximum suppression in a post-processing step [45]. Compared to these works, we aim for common localization in time and space simultaneously.

Closest to our work is spatio-temporal video relocalization by Feng *et al.* [7], which extends their temporal video relocalization [8] to time and space from a *single* support video. Feng *et al.* [7] propose a warp LSTM to align spatio-temporal information between adjacent frames to embed long-term dependencies. They produce a large set of proposals for each query video clip to match each proposal with their single support video. Like Yang *et al.* [45], their method requires hand-designed components in the proposal generation and post-processing step. Compared to spatio-temporal video relocalization, we consider a more general and realistic setting, where more than one support video can be used. We propose a transformer network architecture, which avoids the need for proposals. Lastly, we also generalize to common temporal and pixel-level action localization.

**Transformers in vision.** Transformers were introduced by Vaswani *et al.* [40] as a powerful attention-based building block for machine translation. Jaderberg *et al.* [16] proposed to spatially transform image feature maps, conditional on the feature map itself, alike the self-attention mechanism, even before the standard transformer structure became popular. Gavrilyuk *et al.* [9] propose an actor-transformer to learn and selectively extract information relevant for group activity recognition. Ye *et al.* [47] are the first to employ a self-attention mechanism via a transformer to contextualize instances in the few-shot image classification setting. Recently, Carion *et al.* [3] propose a transformer encoder-decoder structure for object detection to eliminate the need for many hand-designed components, while still demonstrating good performance. Encouraged by this early success of transformers for various vision challenges, we propose a transformer for few-shot common action localization based on the hypothesis that the transformer's attention mechanism
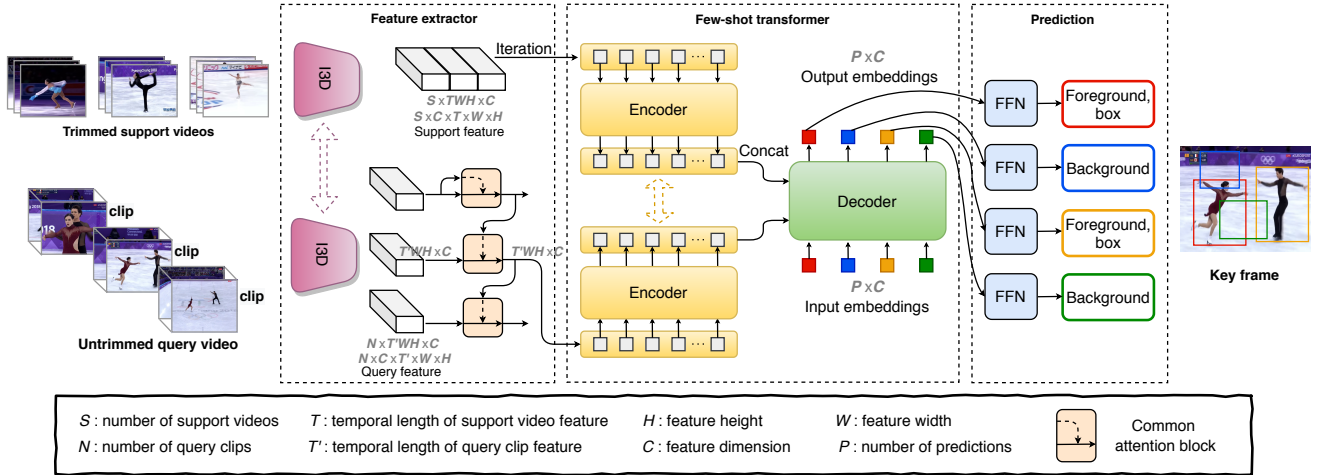
Figure 2: **Overview of the proposed model.** In the feature extraction stage, we obtain the representations of the support videos and the query video clips, and align each query clip feature with previous clips to contain more motion information. In the few-shot transformer, we perform further processing and fusion on the support features and query clip feature. Then we transform the input embeddings to output embeddings by aggregating it with the fused feature. Finally, the output embeddings are used for prediction in the prediction network, which contains a 3-layer feed-forward neural network (FFN) and a linear projection. The detailed structure of the few-shot transformer is depicted in Figure 3.

can efficiently process sequences of spatio-temporal information and model the relation between query and support videos. Our one-stage transformer frees the need for proposals and other pre-defined components, typical in common action localization, while obtaining better localization results in time *and* space.

## 3. Method

Our goal is to localize the spatio-temporal tubelet of an action in an untrimmed query video based on the common action in the trimmed support videos. Note we have no access to the action class label or its time and space boundary annotations. Following recent work on *supervised* spatio-temporal action detection *e.g.*, [11, 20, 46], our approach performs common localization at clip level, *i.e.*, common localization results are first obtained from each query video clip and then linked to build the action tubes across the whole query video. Our approach consists of three main stages. First, we split the untrimmed query video into clips and feed them into the feature extractor, together with the few support videos, to obtain spatio-temporal representations. Each query clip feature is aligned with previous clip features using our common attention block. Then our few-shot transformer further processes the features, fuses the support features into the query clip feature, and transforms the input embeddings by aggregating it with the fused feature. Finally, a 3-layer feed-forward neural network and a linear projection are added on top of the output embeddings from the few-shot transformer to make the final tubelet prediction.

Our method is summarized in Figure 2 and described for each stage in more detail next.

### 3.1. Video feature extractor

We adopt an I3D network [4] as our backbone to obtain spatio-temporal representations of a single query video and a few support videos. The weights of the backbone network are shared between the support and query videos. For the support videos, we feed the whole video into the backbone directly, one by one. Let $f^s \in \mathbb{R}^{S \times TWH \times C}$ denote the representation of all $S$ support videos with their spatio-temporal dimensions flattened, and $f^s_i \in \mathbb{R}^{TWH \times C}$ denote the feature of the $i$-th support video. For a single untrimmed query video, we first split the video into multiple clips. Then the query clips go through the backbone network. Let $f^q \in \mathbb{R}^{N \times T'WH \times C}$ denote the representation of all $N$ query video clips, and $f^q_j \in \mathbb{R}^{T'WH \times C}$ the representation of the $j$-th query clip.

As is, the query clips are separate representations, *i.e.*, $f^q_j$ only contains the spatio-temporal information within the $j$-th query video clip. To accurately localize the tubelets in the query video, long-term spatio-temporal information needs to be modeled. Our common attention block solves this problem. The common attention block is built on the self-attention mechanism [40] and the non-local structure [41]. The self-attention mechanism in the transformer is extended to self- and cross-attention $\mathbb{A}$:

$$\mathbb{A}(I_1, I_2) = \text{softmax}(\frac{Q(I_1)K(I_2)^T}{\sqrt{C}})V(I_2), \quad (1)$$

where $Q(\cdot), K(\cdot), V(\cdot)$ are linear projections. $I_1, I_2$ denote
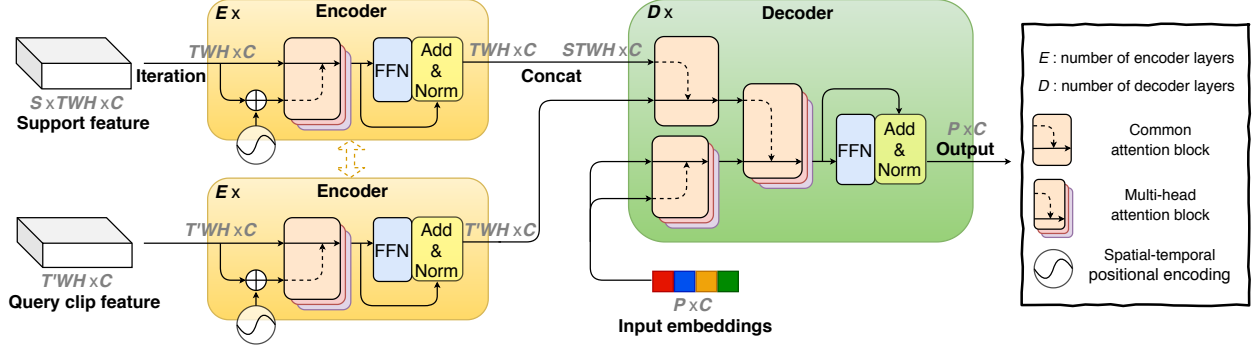
Figure 3: **Structure of few-shot transformer.** The encoder performs further processing on the support features and query clip feature. The decoder fuses the support feature into the query clip feature, then integrate the fused feature into the input embeddings to obtain the output embeddings for the final prediction.

two inputs, both of which have dimension $C$. The common attention block $\widehat{\mathbb{A}}$ can be defined as:

$$\widehat{\mathbb{A}}(I_1, I_2) = I_1 + \text{Linear}(\text{LayerNorm}(\mathbb{A}(I_1, I_2))), \quad (2)$$

where for simplicity, Dropout [37] and ReLU [10] operations are ignored here. A detailed overview and illustration of the common attention block is provided in the supplementary materials. As demonstrated by prior work [14, 26, 45], a self- and cross-attention mechanism can efficiently enhance a representation by itself or by other representations. In our model, the common attention block plays two important roles. In the feature extraction stage, the common attention block aligns each query clip feature with its previous clip features, making each query clip feature contain more motion information, which benefits the common action localization. In the few-shot transformer structure, the common attention block fuses the support feature into the query clip feature, making common action localization achievable.

We utilize the common attention block $\widehat{\mathbb{A}}$ to propagate the spatio-temporal information from previous clips of the query video into the $j$-th query clip for better common localization:

$$\hat{f}_j^q = \widehat{\mathbb{A}}(f_j^q, \hat{f}_{j-1}^q), \quad (3)$$

where $\hat{f}_j^q \in \mathbb{R}^{T'WH \times C}$ is the $j$-th query clip representation enhanced by previous clips.

### 3.2. Few-shot transformer

In the standard transformer, and in our few-shot transformer as well (detailed in Figure 3), the multi-head attention block is a key element. The multi-head attention block $\widehat{\mathbb{MA}}$ is built on multi-head attention $\mathbb{MA}$ which is an extension of attention $\mathbb{A}$:

$$\mathbb{MA}(I_1, I_2) = \text{concat}(\mathbb{A}_1, \cdots, \mathbb{A}_m)W, \quad (4)$$

$$\widehat{\mathbb{MA}}(I_1, I_2) = \text{LayerNorm}(I_1 + \mathbb{MA}(I_1, I_2)), \quad (5)$$

where $m$ denotes the number of heads, $W$ denotes a linear projection, $\mathbb{A}_1, \cdots, \mathbb{A}_m$ are parallel instances of $\mathbb{A}(I_1, I_2)$.

**Encoder.** Each encoder layer has a standard architecture and consists of a multi-head attention block $\mathbb{MA}$ and a feed forward network, FFN. Since the encoder architecture is permutation-invariant, we supplement it with fixed spatio-temporal positional encodings [1, 30] that are added to both support video and query clip features in each encoder layer. For the support branch, we let the support video go through the encoder one by one, then concatenate the output and get a support branch input $E^s \in \mathbb{R}^{STWH \times C}$ to the decoder:

$$\hat{E}(f_i^s) = \widehat{\mathbb{MA}}(f_i^s, pos(f_i^s)), \quad (6)$$

$$E(f_i^s) = \text{LayerNorm}(\hat{E}(f_i^s) + \text{FFN}(\hat{E}(f_i^s))), \quad (7)$$

$$E^s = \text{concat}(E(f_1^s), \cdots, E(f_S^s)), \quad (8)$$

where $S$ denotes the number of support videos and $pos(\cdot)$ denotes the operation of adding spatio-temporal positional encoding, FFN can be seen as a $1 \times 1$ convolutional layer. For the query branch, we let the enhanced query clip feature $\hat{f}_j^q$ go through the encoder and get the query branch input to the decoder $E^q = E(\hat{f}_j^q) \in \mathbb{R}^{T'WH \times C}$.

**Decoder.** In a standard transformer decoder there are two inputs, but in our few-shot transformer decoder there are three inputs: the support branch input $E^s$, the query branch input $E^q$ from the encoder, and the *input embeddings*, which are the learnt positional encodings. For the decoding, we first fuse the support feature into the query clip feature by using the common attention block, *i.e.* $\widehat{\mathbb{A}}(E^q, E^s)$. Then, we perform self-attention on the *input embeddings* via the multi-head attention block, *i.e.* $\widehat{\mathbb{MA}}(in, in)$:

$$\hat{D} = \widehat{\mathbb{MA}}(\widehat{\mathbb{A}}(E^q, E^s), \widehat{\mathbb{MA}}(in, in)). \quad (9)$$

The *input embeddings* are transformed into *output embeddings* by aggregating the fused feature

through another multi-head attention block:

$$out = \text{LayerNorm}(\hat{D} + \text{FFN}(\hat{D})), \qquad (10)$$

which we use as representation for our final prediction.

### 3.3. Prediction network

For final prediction, a 3-layer perceptron is performed on the *output embedding* of the few-shot transformer, with a ReLU activation and hidden dimension $C$, followed by a linear set-based projection layer. The predicted results are the normalized center coordinates, the height and the width of the boxs w.r.t. the keyframe of the $j-$th query clip, along with the *foreground/background* binary labels. Just like Carion *et al.* [3], we use a set-based Hungarian loss [24, 38] that forces unique predictions for each ground-truth bounding box via bipartite matching. Given the clip-level common localization results, we link them over time [20] to obtain the final action tubes for the whole untrimmed query video.

## 4. Experimental setup

Existing video datasets are usually created for other vision tasks, such as classification [15, 23], temporal localization [2], action recognition [36], captioning [5], or summarization [12]. Feng *et al.* [7] reorganize the videos in AVA [11] (version 2.1) into a new dataset for their one-shot spatio-temporal video relocalization task, which is unsuitable for the few-shot setting of our task. To evaluate few-shot spatio-temporal common action localization, we have revised two existing datasets that come with spatio-temporal annotations suitable for our evaluation, namely AVA [11] (version 2.2) and UCF101-24 [36].

**Common-AVA.** There are 430 15-minute video clips with per-second action bounding box annotations in AVA [11]. The annotated actions cover 80 atomic action categories, including "stand","talk", "listen", *etc*. The actions are exhaustively annotated, covering 1.11 million action annotations with multiple labels per person. We link the consecutive bounding boxes if they have the same subject with all the action labels being the same. After linking, the tubelets with exactly the same action labels are regarded as semantically corresponding to each other. Hence, we combine multiple atomic action labels that are annotated with one bounding box together. Next, we discard the combined classes with less than 32 instances and obtain a total of 356 combined classes. We randomly select 80% of the classes for training, 10% of the classes for validation, and the remaining 10% of the classes for testing.

**Common-UCF.** UCF101 [36] is originally an action classification dataset, and a subset of 24 classes with 3,207 videos are provided with spatio-temporal annotations for action detection [36]. We use the recently revised ground truth action tube annotations from Singh *et al.* [34]. In UCF101-24, each action has a single class label. Among the 24 action

| | Common-AVA | Common-UCF |
|---|---|---|
| **Video statistics** | | |
| mean number of instances | 1 | 1.4 |
| mean length (sec) | 4.3 | 7.1 |
| frames per second | 25 | 25 |
| number of train videos | 160,889 | 2,418 |
| number of val+test videos | 12,794 | 776 |
| **Class statistics** | | |
| number of train actions | 286 | 16 |
| number of val+test actions | 70 | 8 |

Table 1: **Overview of the common spatio-temporal localization datasets.** Common-AVA contains a single target action per video, while Common-UCF contains videos with more action instances and longer duration.

categories in UCF101-24, 16 are used for training, 4 for validation and 4 for testing.

During training, the support videos and query video are randomly paired, while the pairs are fixed for validation and testing. Note that we do not use action class labels. More details of the two reorganized datasets are listed in Table 1.

**Implementation details.** We resize all the videos to a $320 \times 320$ resolution before feeding them into the backbone. The I3D model we use is first initialized by training on the Kinetics dataset [23] and then fine-tuned during the training of our model. We initialize the *input embeddings* by setting the values to zeros and we set its dimensionality $P$ to 10. It determines the number of predicted boxes for each keyframe. The encoder and decoder both contain 6 layers. The length of the trimmed support video is 1 second. To form a batch during the training process, the length of the query videos needs to be fixed. The query video is fixed to be 4 seconds long by randomly cropping or padding zeros in the training process. During testing, the query video in full length is fed into the model without any batching and cropping. For Common-AVA the query clip is 1 second long, as a keyframe is annotated per second in the original AVA dataset. For Common-UCF, the query clip is $\frac{1}{5}$ second long because each frame in the videos is annotated at 25 FPS. So for Common-UCF, each query clip contains 5 frames, and we treat the middle frame as the keyframe of the clip.

**Training regime.** We train our model with AdamW [27], setting the initial few-shot transformer's learning rate to $10^{-4}$, the backbone's to $10^{-5}$, and weight decay to $10^{-4}$. We train for 100 epochs with a learning rate drop by a factor of 10 after 70 epochs, where a single epoch is a pass over 10K training videos for Common-AVA and all training videos for Common-UCF. The batch size is set to be 12 with four Nvidia GTX 1080TI cards.

**Evaluation metrics.** We report the frame-level mean average precision (frame-mAP) with an IoU threshold of 0.5 for both datasets. A prediction is correct when it has the cor-

| Encoder | | Decoder | Few-shot setting | |
|---|---|---|---|---|
| support | query | | one-shot | five-shot |
| | | ✓ | 20.3 | 21.9 |
| | ✓ | ✓ | 23.8 | 25.0 |
| ✓ | | ✓ | 22.4 | 24.5 |
| ✓ | ✓ | ✓ | **25.6** | **28.1** |

Table 2: **Few-shot transformer ablation** on Common-AVA. The decoder achieves competitive performance by itself. It improves further when the encoder processes the support and/or query video. Best performance is obtained if the encoder processes both the support and query videos.

| Positional encoding | | Few-shot setting | |
|---|---|---|---|
| support | query | one-shot | five-shot |
| | | 23.3 | 24.5 |
| | ✓ | 24.4 | 25.8 |
| ✓ | | 24.1 | 26.4 |
| ✓ | ✓ | **25.6** | **28.1** |

Table 3: **Benefit of positional encoding.** Adding spatio-temporal positional encoding to the support or query features improves performance on Common-AVA. Including it in both support and query leads to best results.

rect $foreground/background$ prediction and has a ground truth overlap larger than the overlap threshold. On Common-UCF we also report the video-mAP.

# 5. Results

**Few-shot transformer ablation.** We evaluate the effect of the encoder and decoder in our few-shot transformer. We report results for one and five support videos in Table 2. The decoder alone can achieve common spatio-temporal localization and obtains competitive performance. Performance improves notably when we add encoder processing of either support videos or query video. We can gain 5.3 mAP increase under the one-shot setting and 6.2 mAP increase under the five-shot setting, if we let both support and query features go through the encoder.

**Benefit of positional encoding.** We report results in Table 3. Including the spatio-temporal positional encoding is always beneficial, ideally in both the support and query encoder. Not passing any spatio-temporal positional encoding to the features leads to an mAP drop of 2.3 under the one-shot and an mAP drop of 3.6 under the five-shot setting.

**Influence of support video length and amount.** Next, we ablate the effect of the length and the number of support videos on the spatio-temporal localization performance on Common-AVA in Figure 4. We sample 5, 10, 15, 20 and 25 frames for each support video and increase the support video number from 1 to 5. We find that the result gradually increases with longer support videos and/or more support videos, indicating that increasing the support information is beneficial to our modules for common action localization.

**Effect of noisy support videos.** To test the robustness of our approach, we ablate the effect of noisy inputs in the five-shot setting. Video-level noise is simulated by including noisy videos of other actions, or containing no action, in the support videos. Frame-level noise is included by not trimming the supports videos, thus several frames without common action support remain. The results are shown in Table 4. When one out of five support videos contains the
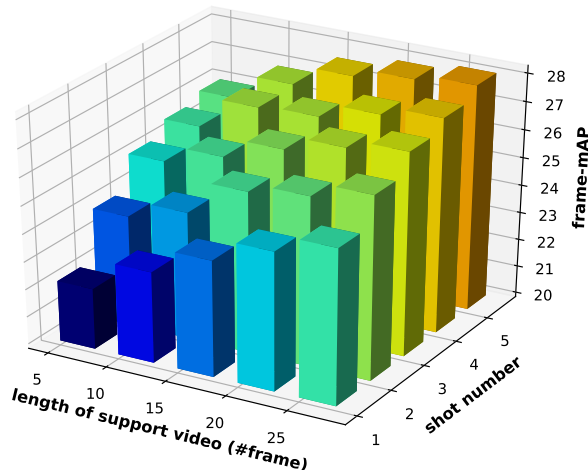


Figure 4: **Influence of length and number of support videos** on Common-AVA. We obtain a more precise common localization with more and longer support videos.

wrong action, the performance drops only 1.3% from 28.1 to 26.8. Where we observe a noisy video containing no action is worse than a video containing a non-common action. When two noisy support video are from the same class, the drop is larger, which is to be expected, as this creates a stronger bias towards a distractor class. Adding a few noisy frames to the (25-frame) trimmed support videos results in slight performance drops, only when 32% of the frames in each support video are noisy, results start to suffer a bit, with a drop from 28.1 to 24.5. Overall, we find that our approach is robust to both video-level and frame-level noise for common action localization in time and space.

**Common action localization in time and space.** In this experiment we demonstrate the effectiveness of our transformer model on the task of common action localization in time and space. As the task is novel, we cannot compare with existing methods that are not designed for this task. Instead, we have adapted existing one-stage and two-stage methods, intended for other tasks. Hu *et al*. [14] and Kang *et al*. [22] provide one-stage approaches for few-shot common object de-
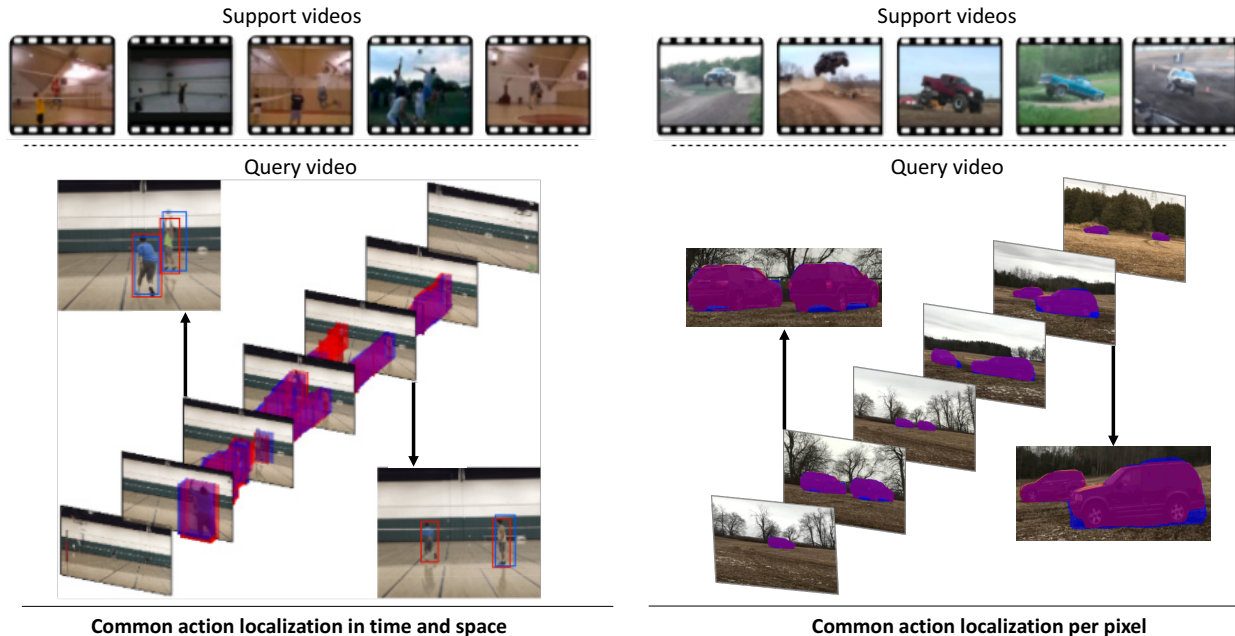
Figure 5: **Qualitative result** under one-shot (blue) and five-shot (red) settings. In the upper part are 2 sets of 5 support videos, where the leftmost video in each set is also used in the one-shot setting. For the left example of common action localization in space and time, with one support video we only miss a short piece at the end of a common action tube. When more support videos are added, we recover the missed clip. For the right example of common action localization per pixel, with one support video we can localize the common action per pixel, and we present more precise localization per pixel with five support videos.

| No noise | 28.1 |
|---|---|
| **Video-level noise** | |
| 1 noisy support video of other class | 26.8 |
| 1 noisy support video without action | 26.3 |
| 2 noisy support videos of different class | 25.3 |
| 2 noisy support videos of same class | 24.7 |
| **Frame-level noise** | |
| 2 noisy frames in each support video | 27.9 |
| 4 noisy frames in each support video | 27.4 |
| 6 noisy frames in each support video | 26.1 |
| 8 noisy frames in each support video | 24.5 |

Table 4: **Effect of noisy support videos** on Common-AVA for the five-shot setting. The result shows our robustness.

tection. We replace their image feature extractors with an I3D network [4] to extract video features and transform the 4D video feature to a 3D feature by flattening along the temporal dimension and the channel dimension. With the flattened features, the few-shot object detector pipelines can localize the bounding box in the keyframe of each query video clip. The spatio-temporal video relocalization of Feng *et al.* [7] is a two-stage approach based on Faster R-CNN [31], which relies on a large set of proposals and contains several pre-

defined components, such as the anchors in the proposal generation and non-maximum suppression in post-processing. Our few-shot transformer frees the need for proposals and such hand-crafted components. Also, where Feng *et al.* [7] is restricted to the one-shot setting, we attempt to make their method compatible with our multi-shot setting. We do so by replacing the matching module of Feng *et al.* [7] with the modules of Yang *et al.* [45], which can efficiently fuse multiple support videos with the query video.

We compare to both baselines under the one- and five-shot setting on Common-AVA and Common-UCF in Table 5. Our approach achieves the best result on both datasets, for both settings. Notably, our performance gains increase more than the baselines as more support videos are available, indicating that our transformer architecture is better able to leverage the commonalities between query and support features. We show a qualitative result in Figure 5 and provide more in the supplemental material.

**Common action localization in time only.** This task strives to localize the temporal extent of a common action in the untrimmed query video [45]. This is a related but different task from ours. Yet, our approach is naturally extended by changing the prediction outputs from a spatial bounding box for a query video clip to a temporal bounding box for the whole query video. We compare to Yang *et al.* [45], Feng *et al.* [8] and Zhang *et al.* [49]. Yang *et al.* [45] introduce a

| | Common-AVA | | Common-UCF | | | |
|---|---|---|---|---|---|---|
| | frame-mAP | | frame-mAP | | video-mAP | |
| | one-shot | five-shot | one-shot | five-shot | one-shot | five-shot |
| Hu *et al.* [14] | 18.4 | 21.1 | 52.6 | 54.9 | 40.7 | 42.9 |
| Kang *et al.* [22] | 19.7 | 21.5 | 54.6 | 56.2 | 42.5 | 43.8 |
| Feng *et al.* [7] | 21.6 | 23.2 | 59.4 | 61.3 | 46.9 | 48.5 |
| *This paper* | **25.3** | **28.1** | **64.3** | **66.7** | **49.8** | **52.7** |

Table 5: **Common action localization in time and space** on Common-AVA and Common-UCF. We constructed two one-stage baselines [14, 22] and a two-stage baseline [7] from author-provided code. Our few-shot transformer obtains favourable results under the one- and five- shot settings on both datasets, without the need for proposals or other pre-defined components.

| | Common-Instance | | Common-Multi-instance | |
|---|---|---|---|---|
| | one-shot | five-shot | one-shot | five-shot |
| Feng *et al.* [8] | 43.5 | n.a. | 31.4 | n.a. |
| Zhang *et al.* [49] | 49.5 | 52.0 | 41.4 | 43.5 |
| Yang *et al.* [45] | 53.1 | 56.5 | 42.1 | 43.9 |
| *This paper* (C3D) | 57.5 | 60.6 | 47.8 | 48.7 |
| *This paper* (I3D) | **59.3** | **61.9** | **50.2** | **52.3** |

Table 6: **Common action localization in time only.** All methods follow the setups of Feng *et al.* [8] and Yang *et al.* [45] on their ActivityNet derived datasets. The metric is video-mAP with an overlap threshold of 0.5. With the same C3D backbone we already outperform all methods, even though our approach is not designed for this setting.

two-stage approach that localizes the common action in time from a few support videos, while Feng *et al.* [8] introduce a one-stage approach for this task intended for a single support video. Zhang *et al.* [49] perform few-shot temporal action localization, where video-level annotations are needed. When we assume the query and support videos share common actions, the one-way few-shot variant of Zhang *et al.* becomes suitable for common action localization in time. We experiment on the reorganized datasets of Yang *et al.* [45], built upon Activity1.3 [2], where the common instance split contains one single action in each video and the common multi-instance split contains multiple actions in each video. Results in Table 6 show results, where all methods use the same C3D [39] backbone. We outperform all three methods on the common temporal localization task, even though our approach is not designed for this setting.

**Common action localization per pixel.** Finally, we demonstrate that our few-shot transformer is also suitable for common action localization per pixel. We consider this specialization of our task to further demonstrate the potential and generalization of our few-shot transformer. We simply add a mask-head on top of the decoder outputs, which predicts a binary mask for each pixel inside the predicted boxes. For evaluation we reorganize the A2D dataset by Xu *et al.* [42], which provides dense pixel-level annotations

| | Common-A2D | |
|---|---|---|
| | one-shot | five-shot |
| Siam *et al.* [33] | 43.3 | 44.8 |
| *This paper* | **50.6** | **52.5** |

Table 7: **Common action localization per pixel** on reorganized A2D dataset. We adapted the few-shot object segmentation of Siam *et al.* [33] to our task. The metric is mIoU across query videos. We perform better across both settings.

for 3,782 videos of 43 actor-action classes. Among the 43 actor-action classes, 33 are used for training, 5 for validation and 5 for testing. The structure of our mask-head, the reorganization of Common-A2D, and the training details for common action localization per pixel are provided in the supplementary material. For comparison we adapt the few-shot common object segmentation approach by Siam *et al.* [33] as our baseline. We replace their flattened image feature sequences with the flattened video feature sequences. We compare to the baseline under one- and five- shot settings on Common-A2D in Table 7. Our approach performs better under both the one- and five- shot settings. In Figure 5 we also show a per-pixel segmentation result.

## 6. Conclusion

We consider spatio-temporal action localization in an untrimmed query video given a few trimmed support videos with a common action, without specifying the action label, temporal bounds, or spatial bounds. To tackle this challenge, we propose a few-shot transformer with a dedicated encoder-decoder structure optimized for joint commonality learning and localization prediction. Evaluation on reorganizations of AVA and UCF101-24 show that our approach localizes a common action in time and space, even when support videos are noisy. Although designed for spatio-temporal localization, our approach generalizes to temporal and per-pixel common action localization, outperforming the respective state-of-the-arts.

# References

[1] Irwan Bello, Barret Zoph, Ashish Vaswani, Jonathon Shlens, and Quoc V Le. Attention augmented convolutional networks. In *ICCV*, 2019.

[2] Fabian Caba Heilbron, Victor Escorcia, Bernard Ghanem, and Juan Carlos Niebles. Activitynet: A large-scale video benchmark for human activity understanding. In *CVPR*, 2015.

[3] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, 2020.

[4] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *CVPR*, 2017.

[5] David L Chen and William B Dolan. Collecting highly parallel data for paraphrase evaluation. In *ACL*, 2011.

[6] Qi Fan, Wei Zhuo, Chi-Keung Tang, and Yu-Wing Tai. Few-shot object detection with attention-rpn and multi-relation detector. In *CVPR*, 2020.

[7] Yang Feng, Lin Ma, Wei Liu, and Jiebo Luo. Spatio-temporal video re-localization by warp lstm. In *CVPR*, 2019.

[8] Yang Feng, Lin Ma, Wei Liu, Tong Zhang, and Jiebo Luo. Video re-localization. In *ECCV*, 2018.

[9] Kirill Gavrilyuk, Ryan Sanford, Mehrsan Javan, and Cees GM Snoek. Actor-transformers for group activity recognition. In *CVPR*, 2020.

[10] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In *AISTATS*, 2011.

[11] 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 *CVPR*, 2018.

[12] Michael Gygli, Helmut Grabner, Hayko Riemenschneider, and Luc Van Gool. Creating summaries from user videos. In *ECCV*, 2014.

[13] Fabian Caba Heilbron, Juan Carlos Niebles, and Bernard Ghanem. Fast temporal activity proposals for efficient detection of human actions in untrimmed videos. In *CVPR*, 2016.

[14] Tao Hu, Pascal Mettes, Jia-Hong Huang, and Cees GM Snoek. Silco: Show a few images, localize the common object. In *ICCV*, 2019.

[15] Haroon Idrees, Amir R Zamir, Yu-Gang Jiang, Alex Gorban, Ivan Laptev, Rahul Sukthankar, and Mubarak Shah. The thumos challenge on action recognition for videos "in the wild". *CVIU*, 2017.

[16] Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. Spatial transformer networks. In *NIPS*, 2015.

[17] Mihir Jain, Amir Ghodrati, and Cees GM Snoek. ActionBytes: Learning from trimmed videos to localize actions. In *CVPR*, 2020.

[18] Mihir Jain, Jan C. van Gemert, Hervé Jégou, Patrick Bouthemy, and Cees G. M. Snoek. Action localization by tubelets from motion. In *CVPR*, 2014.

[19] Mihir Jain, Jan C van Gemert, Thomas Mensink, and Cees GM Snoek. Objects2action: Classifying and localizing actions without any video example. In *ICCV*, 2015.

[20] Vicky Kalogeiton, Philippe Weinzaepfel, Vittorio Ferrari, and Cordelia Schmid. Action tubelet detector for spatio-temporal action localization. In *ICCV*, 2017.

[21] Vicky Kalogeiton, Philippe Weinzaepfel, Vittorio Ferrari, and Cordelia Schmid. Joint learning of object and action detectors. In *ICCV*, 2017.

[22] Bingyi Kang, Zhuang Liu, Xin Wang, Fisher Yu, Jiashi Feng, and Trevor Darrell. Few-shot object detection via feature reweighting. In *ICCV*, 2019.

[23] Will Kay, Joao Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, Mustafa Suleyman, and Andrew Zisserman. The kinetics human action video dataset. *arXiv*, 2017.

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

[25] Tianwei Lin, Xiao Liu, Xin Li, Errui Ding, and Shilei Wen. BMN: Boundary-matching network for temporal action proposal generation. In *ICCV*, 2019.

[26] Weide Liu, Chi Zhang, Guosheng Lin, and Fayao Liu. CR-Net: Cross-reference networks for few-shot segmentation. In *CVPR*, 2020.

[27] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2018.

[28] Pascal Mettes and Cees GM Snoek. Spatial-aware object embeddings for zero-shot localization and classification of actions. In *ICCV*, 2017.

[29] Pascal Mettes, William Thong, and Cees GM Snoek. Object priors for classifying and localizing unseen actions. *IJCV*, 2021.

[30] Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Łukasz Kaiser, Noam Shazeer, Alexander Ku, and Dustin Tran. Image transformer. In *ICML*, 2018.

[31] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NIPS*, 2015.

[32] Zheng Shou, Dongang Wang, and Shih-Fu Chang. Temporal action localization in untrimmed videos via multi-stage cnns. In *CVPR*, 2016.

[33] Mennatullah Siam, Naren Doraiswamy, Boris N Oreshkin, Hengshuai Yao, and Martin Jagersand. Weakly supervised few-shot object segmentation using co-attention with visual and semantic inputs. In *IJCAI*, 2020.

[34] Gurkirt Singh, Suman Saha, Michael Sapienza, Philip HS Torr, and Fabio Cuzzolin. Online real-time multiple spatiotemporal action localisation and prediction. In *ICCV*, 2017.

[35] Khurram Soomro and Mubarak Shah. Unsupervised action discovery and localization in videos. In *ICCV*, 2017.

[36] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. UCF101: A dataset of 101 human actions classes from videos in the wild. *arXiv*, 2012.

[37] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *JMLR*, 2014.

[38] Russell Stewart, Mykhaylo Andriluka, and Andrew Y Ng. End-to-end people detection in crowded scenes. In *CVPR*, 2016.

[39] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In *ICCV*, 2015.

[40] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NIPS*, 2017.

[41] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *CVPR*, 2018.

[42] Chenliang Xu, Shao-Hang Hsieh, Caiming Xiong, and Jason J Corso. Can humans fly? action understanding with multiple classes of actors. In *CVPR*, 2015.

[43] Huijuan Xu, Ximeng Sun, Eric Tzeng, Abir Das, Kate Saenko, and Trevor Darrell. Revisiting few-shot activity detection with class similarity control. *arXiv*, 2020.

[44] Hongtao Yang, Xuming He, and Fatih Porikli. One-shot action localization by learning sequence matching network. In *CVPR*, 2018.

[45] Pengwan Yang, Vincent Tao Hu, Pascal Mettes, and Cees GM Snoek. Localizing the common action among a few videos. In *ECCV*, 2020.

[46] Xitong Yang, Xiaodong Yang, Ming-Yu Liu, Fanyi Xiao, Larry S Davis, and Jan Kautz. Step: Spatio-temporal progressive learning for video action detection. In *CVPR*, 2019.

[47] Han-Jia Ye, Hexiang Hu, De-Chuan Zhan, and Fei Sha. Few-shot learning via embedding adaptation with set-to-set functions. In *CVPR*, 2020.

[48] Gang Yu and Junsong Yuan. Fast action proposals for human action detection and search. In *CVPR*, 2015.

[49] Da Zhang, Xiyang Dai, and Yuan-Fang Wang. METAL: Minimum effort temporal activity localization in untrimmed videos. In *CVPR*, 2020.

[50] Hongyuan Zhu, Romain Vial, and Shijian Lu. TORNADO: A spatio-temporal convolutional regression network for video action proposal. In *ICCV*, 2017.