Context Sensitivity of Spatio-Temporal Activity Detection using Hierarchical Deep Neural Networks in Extended Videos

Felix Hertlein
FZI, Fraunhofer IOSB
hertlein@fzi.de

David Münch
Fraunhofer IOSB
david.muench@iosb.fraunhofer.de

Michael Arens
Fraunhofer IOSB

Abstract

The amount of available surveillance video data is increasing rapidly and therefore makes manual inspection impractical. The goal of activity detection is to automatically localize activities spatially *and* temporally *in a large collection of video data. In this work we will answer the question to what extent context plays a role in spatio-temporal activity detection in extended videos. Towards this end we propose a hierarchical pipeline for activity detection which spatially localizes objects first and subsequently generates spatial-temporal action tubes. Additionally, a suitable metric for performance evaluation is enhanced. Thus, we evaluate our system using the TRECVID 2019 ActEV challenge dataset. We investigated the sensitivity by detecting activities multiple times with various spatial margins around the performing actor. The results showed that our pipeline and metric is suited for detecting activities in extended videos.*

1. Introduction

The task of *spatio-temporal activity detection* is comprised of localizing activities temporally and spatially along with classifying them. Problems hereby are the long time horizon, a varying number of activities, multiple activities simultaneously, arbitrary start and end points, intra- and inter-class variance of the target activities and the computational costs for automatic detection. In the same way there is need for quantitative evaluation, thus a metric for performance evaluation should take the spatial and temporal information into account as well as it should be able to handle long videos with only few activities.

The contributions of this work are: We extended the ActEV 2019 activity detection metric by a spatial component and performed an investigation on the influence of spatial context in spatio-temporal activity detection using Convolutional Neural Networks (CNNs). Furthermore, we developed an entire computer vision pipeline and a visualization tool for dataset and intermediate results analysis.

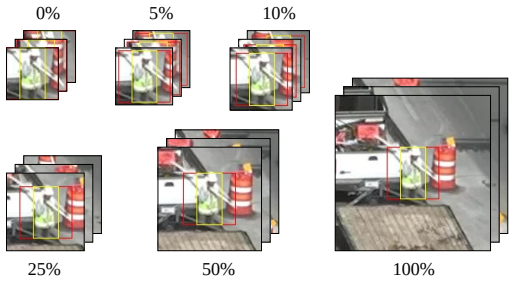

Figure 1. Various spatial contexts relative to the detected object bounding box with a fixed aspect ratio in consecutive frames which form the basis for subsequent temporal activity detection.

2. Related Work

Temporal activity detection. The task of *temporal* activity detection is comprised of localizing activities temporally along with classifying them. Montes *et al*. [14] and Buch *et al*. [3] utilize C3D Networks [18] to generate spatio-temporal features and recurrent neural networks for aggregating said features over longer periods.

A common approach is to generate temporal activity detections in the first place using sliding multi-scale anchors and classify them subsequently [4, 16, 17, 21, 22]. Shou *et al*. [17] propose a solution using three distinct CNNs: Proposal, Classification and Localization. Several works [4, 21, 22] follow the concept and structure of Faster R-CNNs [15] and adapt these to temporal activity detection. Xu *et al*. [22] extend their previous work [21] by integrating an optical flow stream. Chao *et al*. [4] consider multiple receptive fields to cope with the large variation in activity lengths.

Another concept for solving the temporal activity detection problem is the exploitation of dependencies in-between particular activity proposals by modelling proposals and relations as graphs using graph convolution [25].

Spatio-temporal activity detection. The *spatio-temporal* activity detection task requires to find both, spatial and tem-

poral boundaries for all occurrences of pre-specified target activities. One kind of approaches reduces the problem complexity by solving the spatial localization problem first before temporally localizing activities [6, 19, 1, 23, 24, 7]. In early work [6, 19], hand-crafted methods detect frame-wise Regions of Interest (RoI). More recent approaches [1, 23, 24, 7] employ object detection algorithms such as Faster R-CNN [15] to find RoIs. The frame-wise RoIs are combined to tracks over time using different temporal linking methods. Lastly, these approaches employ temporal activity detection methods to determine the temporal boundaries within the tracks. Another approach to spatial-first detection was presented by Hou *et al*. [10]. The authors generate small 3D cuboidal proposals with a fixed size and temporally link them to form longer activity detections.

Opposed to spatial-first detection, Zolfaghari *et al*. [26] propose a multi modal activity detection approach which locates activities first temporally before determining the spatial boundaries. The work of Kalogeiton *et al*. [11] attempts to localize activities both spatially and temporally at once by using 3D anchor cuboids.

3. Proposed Architecture

We propose a hierarchical approach to spatio-temporal activity detection consisting of multiple, consecutive modules. By extracting 3D space-time volumes from the original video, we solve the spatial problem first before temporally localizing activities within said cutouts. We refer to these cutouts as action tubes.

Figure 2 illustrates the pipeline. Our approach consists of five consecutive modules: (1) *Object Detection* localizes and classifies objects in all video frames individually. (2) *Object Tracking* connects the frame-wise detections belonging to the same object in order to form object tracks. (3) *Action Tube Generation* refurbishes object tracks to cope with detection and tracking errors. (4) *Activity Detection* temporally localizes target activities within the action tubes following the general approach of SS-TAD [3]. (5) *Post-Processing* counters the design-related result fragmentation.

3.1. Object Detection

The object detection module takes a sequence of RGB frames $V = \{f_i \in [0,1]^{3\times w\times h}\}_N$ with a total length of N frames and a resolution of $w \times h$. The task of this module is to detect objects in each frame f_i individually, hence it is supposed to generate a set of detections per frame $D_i = \{d_{ij}\}$ with $d_{ij} = (b_{ij}, l_{ij}, c_{ij})$ with a bounding box $b_{ij} = (x_{ij}, y_{ij}, w_{ij}, h_{ij})$, a class label l_{ij} and a class confidence c_{ij} per detection.

For this task, we employ Faster R-CNN with Feature Pyramid Networks [13] with a ResNet-50 backbone [9] using pre-trained weights. For activity detection, we are only interested in activity performing actors such as persons and vehicles. Therefore, we discard all non-actor object detections. Furthermore, we drop all detections with a class confidence $c_{ij} < \theta_{min} \in [0,1]$ to minimize the false detection rate. By default, θ_{min} is 0.6.

3.2. Object Tracking

Given the object detections for the frames f_1 to f_i as $\mathcal{D} = \{D_1, ..., D_i\}$, the object tracking module temporally links associated objects together to object tracks and assigns a unique identifier per track. An object track is denoted by $T_k = (d_{t_1*}, ..., d_{t_m*}, u_k)$ with $t_1 < t_2 < ... < t_m$ and the unique identifier $u_k \in \mathbb{N}$. Note, that a track might miss some detections in-between t_1 and t_m.

DeepSORT [20] is a fast method for multi-object tracking-by-detection based on motion information using a Kalman-Filter and appearance information using deep embeddings. We deployed this algorithm in our pipeline due to its efficient and reliable tracking even with short term occlusions. This tracking algorithm is notably fast, thus capable of tracking object detections in real-time.

3.3. Action Tube Generation

We obtain a set of tracks $\mathcal{T} = \{T_1, ..., T_n\}$ from the object tracking module along with the original video file V and transform these to action tubes. An action tube is a spatio-temporal video excerpt which potentially contains zero, one or multiple activities in any given interval. Ideally, there are no overlapping activities in a single action tube, neither temporal nor spatial. The spatial extent of an action tube is supposed to be of similar size than a contained activity. We define action tubes as in Equation 1. The k-th action tube A_k consists of s consecutive RGB frame excerpts, each scaled to $p \times p$ pixels. By default, p is 224 pixels.

$$\begin{gathered} A_k = (e_1, ..., e_s, u_k) \\ with\ e_i \in [0,1]^{3\times p\times p} \subseteq f_j\ \forall i \in \{1, .., s\},\ u_k \in \mathbb{N} \end{gathered} \tag{1}$$

For the transformation, we apply a series of processing steps: gap interpolation, temporal and spatial expansion, smoothing, aspect ratio enforcement and image padding.

Gap Interpolation. We interpolate any gap inside the tracks to fix detection or tracking errors through short occlusions. For a given track T_k with a gap of length z we linearly interpolate the closest available detection bounding boxes b_i and $b_{(i+z+1)}$.

Temporal Expansion. An object might be detected not until a few frames after the emerge the object, especially if the object moves in or leaves the scene. Therefore, the resulting track potentially is too short in the temporal dimension. In order to ensure, that activities are completely captured by action tubes, we add a temporal margin of ϕ frames to each track by linear extrapolation of the outer most two frames.

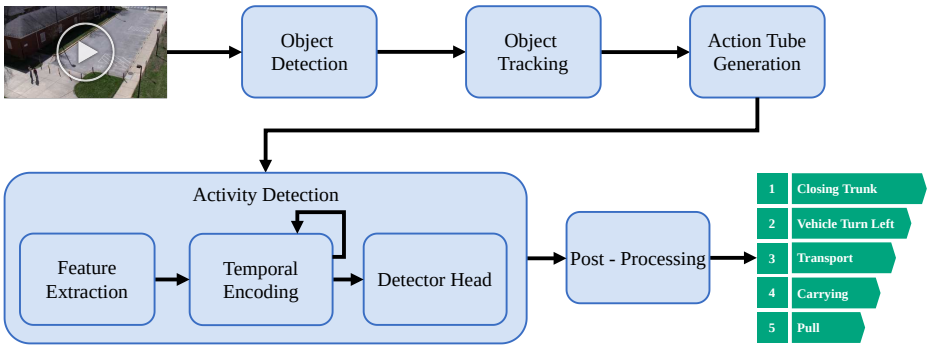

Figure 2. Illustration of the proposed activity detection architecture. The pipeline consists of five consecutive modules: Object Detection, Object Tracking, Action Tube Generation, Activity Detection and Post-Processing.

Spatial Expansion. In general, activities cover a slightly larger area per frame than the size of the detected objects since the interaction objects requires some extra space. In order to increase the context information around detected objects and to cover activities entirely, we extend all bounding boxes in all tracks by a spatial margin of ω percent with respect to the largest side length.

Smoothing. Since the object detections were generated frame-by-frame, consecutive bounding boxes within a track might differ abruptly in position, size and aspect ratio. In contrast, activities have a smooth bounding box transition from frame to frame. In order to fix the discontinuous shape, we apply a linear filter for interpolation between consecutive frames with a smoothing factor γ.

Aspect Ratio Enforcement. For the subsequent processing modules an action tube is required to have a fixed spatial extend of $p \times p$ pixels per frame excerpt e_i. To preserve the aspect ratio of each bounding box, we extend each bounding box individually by increasing the smaller edge length.

Image Padding. On account of the track modifications described in the previous sections, the bounding boxes can stretch out over the image borders. In those cases, we pad the original image by replicating the outer most pixel border until all bounding boxes are entirely contained by the extended image. Thereafter, the image extracts e_i are generated by cutting out b_i from the extended image and scaled to $p \times p$ pixels using bicubic interpolation.

Finally, the action tubes $A_k = (e_1, ..., e_s, u_k)$ are constructed using the identifier u_k supplied by the original track, see Figure 3.

3.4. Activity Detection

The activity detection takes in a set of action tubes $\mathcal{A} = \{A_1, ..., A_r\}$ and temporally localizes activities within those. An activity detection comprises of the start and end frame number f_s, f_e, the activity label l and the prediction confidence c. We follow the general approach of Single Stream Temporal Activity Detection [3]. The temporal activity detection consists of three submodules (see Figure 4): (1) *Feature Extraction* derives spatio-temporal features from short clips (2) *Temporal Encoding* accumulates said features over longer periods (3) *Detector Head* finally detects activities.

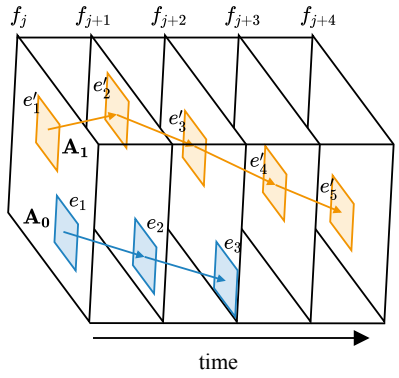

Figure 3. Visualization of exemplary action tubes (yellow and blue). Each tube consists of a sequence of consecutive rectangular frame excerpts with varying position, size and aspect ratio.

3.4.1 Feature Extraction

Given an action tube $A_k = (e_1, ..., e_s, u_k)$, the feature extraction module subdivides the action tube into short clips of d frames. By default, d is 16. For an action tube with length s, the number of clips $n \in \mathbb{N}$ is $\lfloor \frac{s}{d} \rfloor$, see Figure 5. After clip generation an action tube consists of a list of clips as in Equation 2:

$$A_k = (z_1, ..., z_n, u_k) \quad with \quad z_i \in [0,1]^{d \times p \times p \times 3} \tag{2}$$

For upstream temporal encoding, we reduce the dimensionality of each clip z_i by mapping them to low-dimensional spatio-temporal features using a pretrained 3D

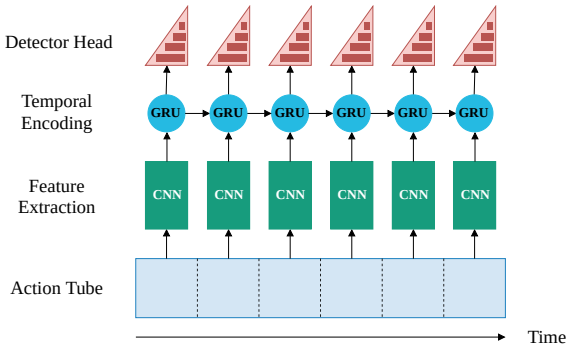

Figure 4. Overview of temporal activity detection modules and data flow in-between over time. Figure is based on the SS-TAD approach from [3].

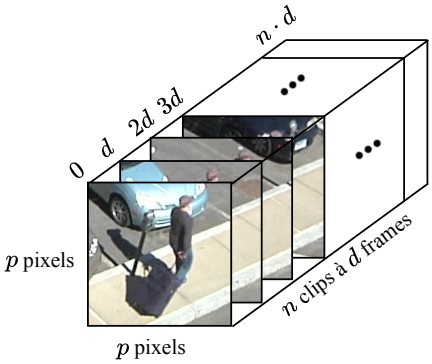

Figure 5. Subdivided action tube in clips of length d. The tube was subdivided into n clips of length d frames. All frames within the clips are scaled to $p \times p$ pixels.

Resnet-34 network [8]. Optimally, these spatio-temporal features encode information about objects and their movements within the d frames long and $p \times p$ wide clip. We feed each clip z_i individually in the pre-trained Resnet-34 and perform a forward inference on them. Through reading the output of the penultimate layer, we obtain 512-dimensional features. After the clip transformation the action tubes $\mathcal{A} = \{A_1, ..., A_r\}$ can be represented as in Equation 3:

$$A_k = (q_1, ..., q_n, u_k) \quad with \quad q_i \in \mathbb{R}^{512} \tag{3}$$

3.4.2 Temporal Encoding

Based on the encoded action tubes from the previous processing step $A_k = (q_1, ..., q_n, u_k)$, we employ stacked Gated Recurrent Units (GRUs, [5]) in order to accumulate relevant information regarding object appearances and movements over a longer period, see Figure 6. We feed in the sequence of clip encodings q_1 to q_n and receive

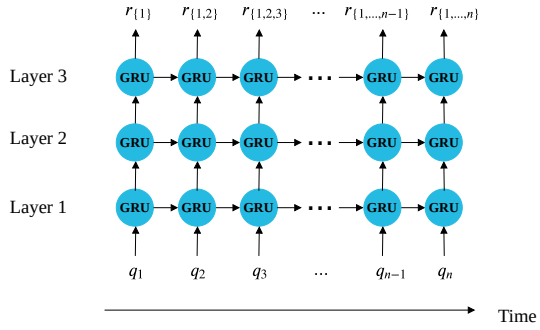

Figure 6. Stacked GRU architecture. The clip encodings generated by the Resnet-34 are accumulated using multiple layers of gated recurrent units.

a sequence of temporal-encodings $r_{\{1\}}$ to $r_{\{1,...,n\}}$ with $r_{\{*\}} \in \mathbb{R}^{1024}$. Note, that each feature vector $r_{\{*\}}$ is dependent to all previously fed encodings. Equation 4 expresses the representation of an action tube after the temporal encoding module.

$$\hat{A}_k = (r_{\{1\}},\ r_{\{1,2\}},\ r_{\{1,2,3\}},\ ...,\ r_{\{1,..n\}},\ u_k) \quad with \quad r_{\{*\}} \in \mathbb{R}^{1024} \tag{4}$$

3.4.3 Detector Head

The detector head module takes in action tubes $\hat{A}_k$ as given by the temporal encoding module and outputs activity detections within the action tubes. Similar to [3], we define τ anchors of varying length in order to localize activities temporally through a single forward pass. An anchor a_i with $i \in \{1, ..., \tau\}$ spans a length of $i \cdot d$ frames for a clip length of d frames. All anchors are aligned in such a manner, that they end at the same frame. Figure 7 shows the alignment (left side).

The detector head is a fully-connected neural network, which infers the activity presence probabilities per anchor and class. By applying the detector head to a given clip position, the anchors correspond to a temporal span within the action tube. The network takes a single feature $r_{\{*\}} \in \mathbb{R}^{1024}$ provided by the temporal encoding module and outputs a matrix $O \in [0, 1]^{\tau \times (c+1)}$ for a total of c activities. The matrix has $c + 1$ columns due to one additional background class, which allows the network to signal no activity for the given temporal span. Each row in the output matrix corresponds to one anchor as defined above. See Figure 7 for the correspondence. The output matrix O expresses the activity presence probability per considered interval and class due to a final softmax layer at the end of the detector head network.

By applying argmax to each row, we obtain the most likely class per interval $M \in \mathbb{N}^{\tau}$. Through applying row-wise the max function, we receive the associated confidence

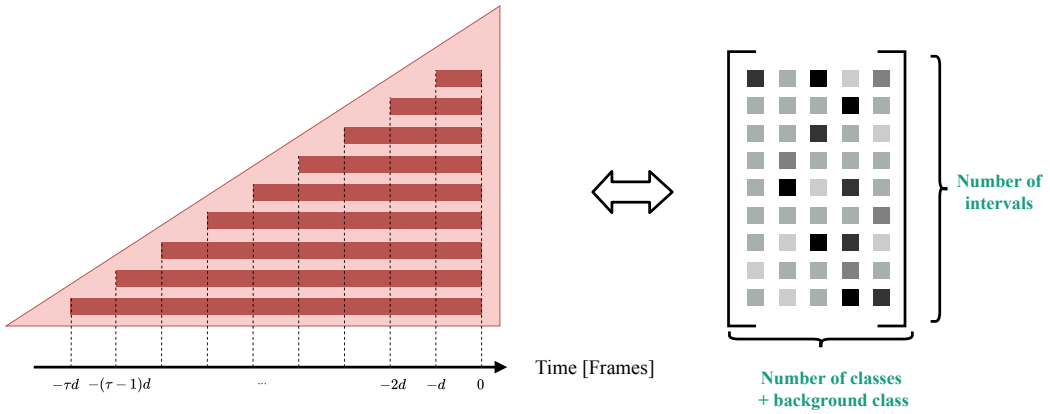

Figure 7. Visualization of the detector head output. The output matrix (right side) encodes class probabilities for all activity classes plus one additional background class. Each row in the output matrix corresponds to a predefined anchor, which defines a time segment relative to the current head position. The predictions are generated with respect to this time segments.

score $P \in [0,1]^{\tau}$ to the most likely class. See Equations 5 and 6.

$$M = \underset{j \in \{1,\ldots,c+1\}}{\operatorname{argmax}} \; o_{ij} \tag{5}$$

$$P = \underset{j \in \{1,\ldots,c+1\}}{\max} \; o_{ij} \tag{6}$$

Since our goal is to detect all activity occurrences, we can ignore all intervals in which the background class is the most likely class. Additionally, we introduce a minimum confidence threshold Θ_{conf} to ensure that detections with low certainty are discarded. By default, Θ_{conf} is 0.5. The temporal activity boundaries are trivially retrieved by adding the corresponding anchor boundaries to the current clip position. The spatial activity boundaries are retrieved by the spatial extent of each action tube minus the applied spatial padding.

Formally, we denote the detections generated by the system as $\mathcal{S} = \{S_1, ..., S_h\}$ with $S_i = (f_{si}, f_{ei}, l_i, c_i, u_i, B_i)$. f_{si}, f_{ei} denote the start and end frame of the i-th detection, l_i the class label, c_i the presence confidence of the system, u_i the action tube identifier and the frame-wise object localizations $B_i = \{b_{ij} \mid j \in \{f_{si}, ..., f_{ei}\}\}$.

Detector Head Architecture. We constructed the detector head architecture as shown in Figure 8. The network is relatively shallow with only three fully connected layers.

Ground Truth Mapping. The ground truth activities are provided as a list of activities in global coordinates. Since the training of the temporal encoding and the detector head module is performed on action tube level, we need to map the ground truth activities to the action tubes $\mathcal{A} = \{A_1, ..., A_r\}$. This is achieved by intersecting all clips $z_i \; \forall i$ with all ground truth activities and assigning the ground truth class with the highest overlap to z_i. If all intersection overlaps are below a certain threshold, the label *background* is assigned.

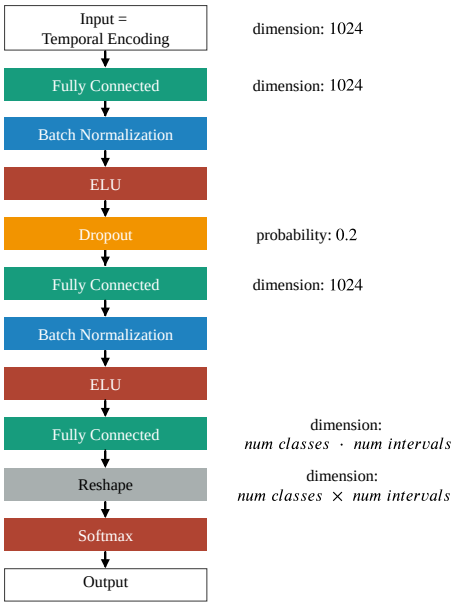

Figure 8. Visualization of the detector head architecture. Three fully connected layer in conjunction with the ELU activation function map the temporal encoding to the output matrix O containing class probabilities.

3.4.4 Training

For joint training of the temporal encoding module and the detector head we utilize a loss function similar to [17] and [3]. This loss consists of two different loss terms: $\mathcal{L}_{detcls}$ penalizes wrong class predictions and $\mathcal{L}_{detloc}$ penalizes minor temporal overlaps between ground truth and considered

interval. Equation 7 shows the combined loss function, whereas λ controls the influence of the overlap loss. By default, λ is 1.

$$\mathcal{L} = \mathcal{L}_{detcls} + \lambda \cdot \mathcal{L}_{detloc} \tag{7}$$

The $\mathcal{L}_{detcls}$ loss penalizes erroneous classification through a weighted log loss (Equation 8). The weights w_i with $i \in \{1, ..., \tau\}$ counter the class-background imbalance problem. Due to the sparsity of activity instances in source videos, we reduce the influence of background classifications according to their relative frequency of occurrence within the entire dataset. The weight w_i is set to 1 if the ground truth of anchor a_i is any non-background class. Otherwise, w_i is the ratio of class instances to background instances for all sliding windows with given anchor length.

$$\mathcal{L}_{detcls} = -\frac{1}{\tau} \sum_{i}^{\tau} w_i \cdot log\left(P_i\right) \tag{8}$$

The $\mathcal{L}_{detloc}$ loss penalizes anchors with minor overlap to ground truth instances. Equation 9 describes the loss computation. v_i refers to the normalized temporal intersection over union between (tIoU) the i-th anchor and the associated ground truth activity $gt(a_i)$. The intersection with a_τ leads to an optimal tIoU value of 1. The operator $\mathbb{1}$ denotes the Iverson bracket and $gt(a_i)$ evaluates to the ground truth class label for the i-th anchor.

$$\begin{gathered} \mathcal{L}_{detloc} = \frac{1}{2} \sum_{i}^{\tau} \left(\frac{o_{ij}^2}{v_i^\alpha} - 1 \right) \mathbb{1}\left[gt(a_i) \neq backgr.\right] \\ v_i = \text{tIoU}(\text{intersection}(gt(a_i), a_\tau), a_i) \end{gathered} \tag{9}$$

3.5. Post-Processing

By construction, our system generates heavily overlapping detections of variable length. To eliminate partial detections, we compare the system instances $\mathcal{S} = \{S_1, ..., S_h\}$ pairwise and merge any two instances $S_i = (f_{si}, f_{ei}, l_i, c_i, u_i, B_i)$ and $S_j = (f_{sj}, f_{ej}, l_j, c_j, u_j, B_j)$ if the following conditions are fulfilled:

Class congruence. Class types must be identical, therefore $l_i = l_j$.

Temporal proximity. Both detections must overlap temporally. The relative temporal overlap $l_{ij} \in [0, 1]$ is defined as in Equation 10. For merging, the relative temporal overlap l_{ij} has to exceed a given threshold $\Theta_{minTIOU}$. By default, $\Theta_{minTIOU}$ is 0.1.

$$l_{ij} = \frac{\text{intersection}([f_{si}, f_{ei}], [f_{sj}, f_{ej}])}{\min(f_{ei} - f_{si}, f_{ej} - f_{ej})} \tag{10}$$

Spatial proximity. Both detections must occur in the same area regarding the spatial dimension. To ensure this property, we enforce to both detections to be originated from the same action tube. Thus, we require the condition $u_i = u_j$.

If all three conditions are met, we merge S_i and S_j and obtain S' from Equation 11.

$$\begin{aligned} S' = (\, &\min(f_{si}, f_{sj}),\ \max(f_{ei}, f_{ej}),\ l_i, \\ &\max(c_i, c_j),\ u_i,\ B_i) \end{aligned} \tag{11}$$

4. Evaluation

We explore the context sensitivity of spatio-temporal activity detection in extended videos using the TRECVID 2019 ActEV challenge dataset [2].

4.1. Metric Extension

Our evaluation metrics follow the evaluation metrics of the ActEV 2019 challenge [2]. In contrast to their metric for activity detection (AD), we additionally include spatial information to create a meaningful metric to measure the quality of a spatio-temporal activity detection system. This allows us to evaluate not only temporally but also spatially the results at activity level. The ActEV 2018 evaluation metrics AOD and AODT include spatial detection information as well, but these metrics depend on the localizations of involved objects. However, our extended metric considers localizations of activities as a whole.

4.1.1 Alignment between ground truth and detections

The evaluation metrics require a one-to-one alignment between system detections $\mathcal{S} = \{S_1, ..., S_h\}$ and ground truth references $\mathcal{R} = \{R_1, ..., R_g\}$. A system detection S_i is either a correct detection (CD) if there is a correspondence R_j in the ground truth activities. Otherwise, S_i is labeled a false alarm (FA). Reference activities are labels either correct detection (CD) or missed detection (MD) depending on the availability of a correspondence.

Two activity instances $S_i = (f_{si}, f_{ei}, l_i, c_i, u_i, B_i) \in \mathcal{S}$ and $R_j = (f_{sj}, f_{ej}, l_j, u_j, B_j) \in \mathcal{R}$ need to fulfill the following criteria in order to be considered at all for an association: class congruence, temporal overlap and spatial overlap. For class congruence, class types must be identical, therefore $l_i = l_j$. Sufficient temporal overlap is given if the tIoU is greater than 0.5 (for system instances < 1 second) and if the intersection is greater than one second (for system instances ≥ 1 second).

The third criterion is the spatial overlap between both instances by a threshold ϵ according to the average spatial intersection over union (Equation 12). By default, ϵ is 0.01.

$$\begin{gathered} \text{asIoU}(S_i, R_j) = \frac{\text{IoU}(B_i[f], B_j[f])}{\sum_f 1} \\ \forall f \in \text{intersection}([f_{si}, f_{ei}], [f_{sj}, f_{ej}]) \end{gathered} \tag{12}$$

To generate the mapping, we utilize the Hungarian algorithm on the bipartite graph matching problem [12]. Activity instances of both kinds are modeled as nodes within a bipartite graph. All system nodes are connected with all reference nodes which qualify as a possible correspondence and vice versa. The edge weights between two nodes S_i and R_j are determined by the system activity presence confidence c_i.

4.1.2 Probability of missed detection

Based on the one-to-one alignment between system instances $\mathcal{S}$ and references $\mathcal{R}$, the metric *probability of missed detection* $\mathrm{P}_{miss}@X\mathrm{rfa}$ is defined as the fraction of missed detection over the total number of ground truth instances at a false alarm rate per minute of X (Equation 13).

$$\mathrm{P}_{miss}(class)@X\mathrm{rfa} = \frac{\#MD_{class}@X\mathrm{rfa}}{\#\mathcal{R}_{class}} \in [0,1] \qquad \mathcal{R}_{class} = \{R_j | R_j \in R \wedge l_j = class\} \tag{13}$$

The false alarm rate per minute X is defined in dependency of a presence confidence threshold Θ_p. For a given Θ_p, the scorer filters the system instances using this threshold and computes an alignment based on the remaining instances. The resulting number of false alarms is divided by the total length of all videos in minutes. By variation of the presence confidence threshold, we thus can calculate different false alarm rate values. Using these values, we can calculate the probability of missed detections at various false alarm rates.

The mean probability of missed detections Mean-P_{miss} combines the $\mathrm{P}_{miss}(class)@X\mathrm{rfa}$ of all activity classes by averaging them.

4.1.3 Normalized Partial Area under DET Curve

The normalized, partial area under detection-error curve reduces the $\mathrm{P}_{miss}(class)@X\mathrm{rfa}$ metric to a single score value by integrating over the false alarm rate up to a maximal false alarm rate. Equation 14 formalizes the nAUDC metric in dependency of the maximal false alarm rate a and a given activity class.

$$\mathrm{nAUDC}_a(class) = \frac{1}{a}\int_0^a \mathrm{P}_{miss}(class)@x\mathrm{rfa}\, dx \tag{14}$$

The mean normalized, partial area under detection-error curve metric combines the $\mathrm{nAUDC}_a(class)$ for all classes by averaging them. A perfect system reaches a Mean-nAUDC_a value of 0. If the system does not detect any activity correctly below the false alarm rate threshold a, the metric evaluates to 1.

4.2. Experiments

For our experiments, we joined the train and validation splits of the TRECVID 2019 ActEV challenge evaluation dataset [2] and utilized 5-fold cross validation for evaluation. The conjoined dataset consists of 2466 activities in total. Since the videos contain multiple activities per file, we employ the grouped K-fold strategy to split the data on activity level. We ran our experiments once per split and average the evaluation results at the end.

In order to investigate the context sensitivity of spatio-temporal activity detection, we measured the performance of our system for various spatial margins specified in the *ActionTubeGeneration* module. The change in spatial margin leads to broader action tubes, which incorporate more context around the activity actor. We evaluate the system for a margin of 0, 5, 10, 25, 50 and 100 percent. Figure 1 visualizes the effect of different margins on the resulting action tubes.

4.2.1 Results

The results for this experiment are shown in Table 1. Since we want to capture the influence of spatial context on the activity detection quality, we consider the spatial margin of 0 as our baseline for comparison. This setting relates to action tubes with no additional context. Thus, they contain only the tracked object.

Spatial margin	Mean-$\mathrm{nAUDC}_{0.2}$
0	0.889098
5	0.872223
10	0.859502
25	**0.854712**
50	1.000000
100	0.994454

Table 1. Results of the context sensitivity experiment. The best activity detection results with a Mean-nAUDC_a of 0.854712 were achieved with a spatial margin of 25%.

The experiments with margins 5, 10 and 25% show an improvement compared to the baseline. This improvement indicates that the close proximity to the activity actor contains activity specific information, which helps our system to locate and classify them. The margin of 25% yields the best performance over all tested margins with a Mean-$\mathrm{nAUDC}_{0.2}$ of 0.854712. Further experiments with the margins 50 and 100% reveal the limitations of spatial action tube padding. At some point in-between 25 and 50% the advantage of a larger context does not outweigh the disadvantage of information loss through scaling. For large margins, the object tracks are large with respect to their spatial dimensions. In the *Activity Generation* module, we scale the action tubes to a fixed spatial resolution of $p \times p$

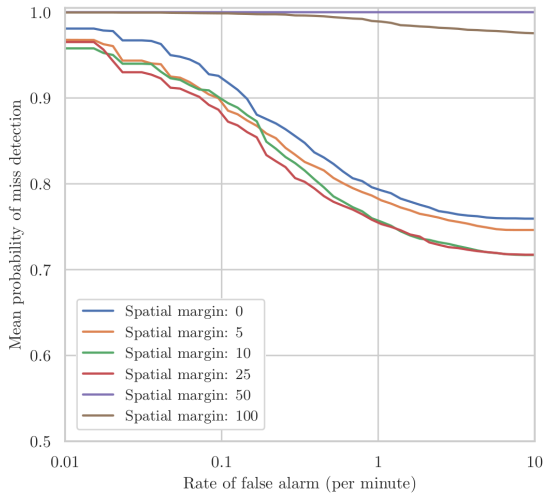

Figure 9. Results of the context sensitivity experiment as an average of all activities for the spatial margins 0, 5, 10, 25, 50 and 100 %.

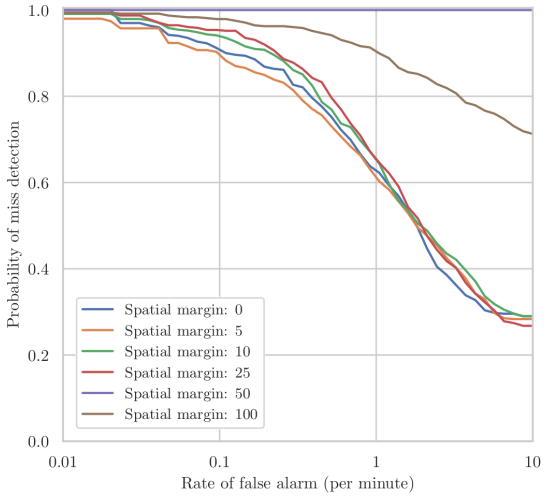

Figure 10. Results of the context sensitivity experiment for the activity *Activity_carrying* for the spatial margins 0, 5, 10, 25, 50 and 100%.

pixels. The loss in information through scaling may be the reason for worse results with higher margins.

Figure 9 displays the same results as a function of Mean-P$_{miss}$ in dependency of the false alarm rate for different spatial margins.

The Figures 10 and 11 demonstrate the dependency of the context sensitivity from the activity type. The prior displays the activity *Activity_carrying*, whereas the latter expresses the activity *Pull*.

In the *Activity_carrying* figure, the lower margins (0, 5, 10 and 25%) perform roughly similar with no clear best option. This circumstance indicates that for the activity *Activity_carrying* additional context information has no or almost no effect on the activity detection quality. This finding is not surprising due to the fact, that carried objects in most cases are kept close to the actor and therefore are already included in the action tubes at a spatial margin of 0. The larger margins (50 and 100%) produce significantly worse results than the smaller margins. This circumstance might be due to the information loss as described above.

Figure 11 shows the probability of missed detection for the activity *Pull*. The effect of adding context to action tubes is substantial. The lower margins (0, 5, 10 and 25%) show a significant performance improvement compared to the baseline. The reason is possibly that the activity *Pull* involves pulling an object, which in general is close to the actor, but not within the same bounding box around the actor. Adding a margin incorporates the pulled object in the action tube and therefore is included into the further pipeline. Again, the larger margins (50 and 100%) produce worse results than the baseline.

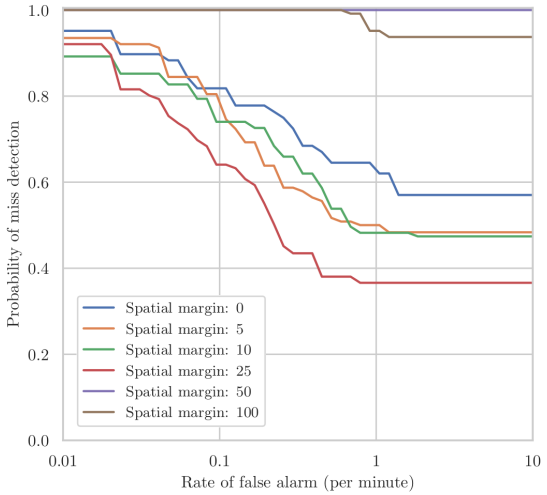

Figure 11. Results of the context sensitivity experiment for the activity *Pull* for the spatial margins 0, 5, 10, 25, 50 and 100%.

5. Conclusion

In this work we developed a hierarchical pipeline for activity detection which spatially localizes objects first and subsequently generates spatial-temporal action tubes. The ActEV 2019 challenge metric was extended by a spatial component and an investigation on the influence of spatial context in spatio-temporal activity detection using CNNs was performed. The results showed that spatial context is class dependent and plays a significant role in detecting activities in extended videos.

References

[1] S. Aakur, D. Sawyer, and S. Sarkar. Fine-grained Action Detection in Untrimmed Surveillance Videos. In *WACV Workshops*, 2019.

[2] ActEV: Activities in Extended Video. `https://actev.nist.gov`, 2019. [Online; accessed 13-June-2019].

[3] S. Buch, V. Escorcia, B. Ghanem, L. Fei-Fei, and J. C. Niebles. End-to-End, Single-Stream Temporal Action Detection in Untrimmed Videos. In *BMVC*, 2017.

[4] Y.-W. Chao, S. Vijayanarasimhan, B. Seybold, D. A. Ross, J. Deng, and R. Sukthankar. Rethinking the Faster R-CNN Architecture for Temporal Action Localization. In *CVPR*, 2018.

[5] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. *arXiv preprint arXiv:1406.1078*, 2014.

[6] G. Gkioxari and J. Malik. Finding Action Tubes. In *CVPR*, 2015.

[7] J. Gleason, R. Ranjan, S. Schwarcz, C. Castillo, J.-C. Chen, and R. Chellappa. A Proposal-Based Solution to Spatio-Temporal Action Detection in Untrimmed Videos. In *WACV*, 2019.

[8] K. Hara, H. Kataoka, and Y. Satoh. Can Spatiotemporal 3D CNNs Retrace the History of 2D CNNs and ImageNet? In *CVPR*, 2018.

[9] K. He, X. Zhang, S. Ren, and J. Sun. Deep Residual Learning for Image Recognition. In *CVPR*, 2016.

[10] R. Hou, C. Chen, and M. Shah. Tube Convolutional Neural Network (T-CNN) for Action Detection in Videos. In *ICCV*, 2017.

[11] V. Kalogeiton, P. Weinzaepfel, V. Ferrari, and C. Schmid. Action Tubelet Detector for Spatio-Temporal Action Localization. In *ICCV*, 2017.

[12] H. W. Kuhn. The Hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97, 1955.

[13] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie. Feature Pyramid Networks for Object Detection. In *CVPR*, 2017.

[14] A. Montes, A. Salvador, S. Pascual, and X. Giro-i Nieto. Temporal Activity Detection in Untrimmed Videos with Recurrent Neural Networks. *arXiv preprint arXiv:1608.08128*, 2016.

[15] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In *NIPS*, 2015.

[16] P. Schlosser, D. Münch, and M. Arens. Investigation on Combining 3D Convolution of Image Data and Optical Flow to Generate Temporal Action Proposals. In *CVPR Workshops*, 2019.

[17] Z. Shou, D. Wang, and S.-F. Chang. Temporal Action Localization in Untrimmed Videos via Multi-stage CNNs. In *CVPR*, 2016.

[18] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri. Learning Spatiotemporal Features with 3D Convolutional Networks. In *ICCV*, 2015.

[19] P. Weinzaepfel, Z. Harchaoui, and C. Schmid. Learning to track for spatio-temporal action localization. In *ICCV*, 2015.

[20] N. Wojke, A. Bewley, and D. Paulus. Simple Online and Realtime Tracking with a Deep Association Metric. In *ICIP*, 2017.

[21] H. Xu, A. Das, and K. Saenko. R-C3D: Region Convolutional 3D Network for Temporal Activity Detection. In *ICCV*, 2017.

[22] H. Xu, A. Das, and K. Saenko. Two-Stream Region Convolutional 3D Network for Temporal Activity Detection. *arXiv preprint arXiv:1906.02182*, 2019.

[23] L. Yao and Y. Qian. DT-3DResNet-LSTM: An Architecture for Temporal Activity Recognition in Videos. In *Pacific Rim Conference on Multimedia*, 2018.

[24] L. Yao and Y. Qian. Novel Activities Detection Algorithm in Extended Videos. In *WACV Workshops*, 2019.

[25] R. Zeng, W. Huang, M. Tan, Y. Rong, P. Zhao, J. Huang, and C. Gan. Graph Convolutional Networks for Temporal Action Localization. In *ICCV*, 2019.

[26] M. Zolfaghari, G. L. Oliveira, N. Sedaghat, and T. Brox. Chained Multi-stream Networks Exploiting Pose, Motion, and Appearance for Action Classification and Detection. In *ICCV*, 2017.