APT: Action localization Proposals from dense Trajectories

Jan C. van Gemert[1,2]
j.c.vangemert@gmail.com
Mihir Jain[1]
m.Jain@uva.nl
Ella Gati[1]
ellagati@gmail.com
Cees G. M. Snoek[1,3]
cgmsnoek@uva.nl

[1] Intelligent Systems Lab Amsterdam
University of Amsterdam
Amsterdam, The Netherlands

[2] Computer Vision Lab
Delft University of Technology
Delft, The Netherlands

[3] Qualcomm Research Netherlands
Amsterdam, The Netherlands

Abstract

This paper is on action localization in video with the aid of spatio-temporal proposals. To alleviate the computational expensive segmentation step of existing proposals, we propose bypassing the segmentations completely by generating proposals directly from the dense trajectories used to represent videos during classification. Our Action localization Proposals from dense Trajectories (APT) use an efficient proposal generation algorithm to handle the high number of trajectories in a video. Our spatio-temporal proposals are faster than current methods and outperform the localization and classification accuracy of current proposals on the UCF Sports, UCF 101, and MSR-II video datasets.
Corrected version: we fixed a mistake in our UCF-101 ground truth. Numbers are different; conclusions are unchanged.

1 Introduction

The goal of this paper is to arrive at localization of actions in video that is faster *and* better than alternatives. Recent action localizations [9, 17, 30, 33] are inspired by successful object-proposals in static images [1, 15, 25]. Compared to sliding-subvolume approaches [10, 19, 22], proposals for action localization reduce the video search space to a small set of spatio-temporal tubes, with high likelihood to contain an action. Different from these existing approaches, which all use different representations for localization and classification, we start from the observation that we can perform both tasks with one and the same representation.

Diving deeper into two popular proposal algorithms [9, 17], three common steps are identified: 1. Pre-processing the video by segmentation, 2. Generating tube proposals by grouping segments and 3. Representing each tube with dense trajectory features [27, 28] so it can be fed to an action recognition classifier. The video segmentation (step 1) easily takes several minutes for a modest video of 400x720 pixels of 55 frames [17, 32] and can take days for a realistic full HD movie. Moreover, a unique meaningful pixel grouping is a more complex problem than necessarily required for proposal generation. The segmentation step for action proposals is overly complex and not practical for large video sets.

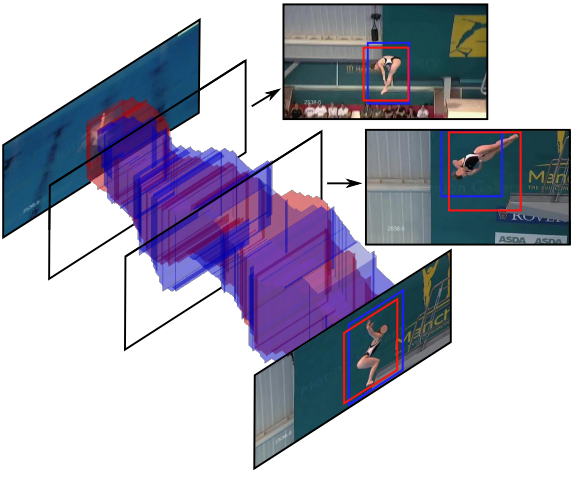

Figure 1: **We propose APT:** an efficient and effective spatio-temporal proposal algorithm for action localization. Different from existing work, which consider different representations for the localization and classification stage, we aptly re-use the dense trajectories as used in the classification representation for localization as well. This allows large-scale deployment with good localization and classification accuracy. For the action *Diving* our blue best proposal results in an overlap of 0.62 with the red ground truth.

We propose to skip the video segmentation of step 1 altogether. Instead, we apply the proposal generation of step 2 on the dense trajectories of step 3. Since the dense trajectories are computed for the representation anyway, we propose to re-use them for proposal generation as well. By founding the proposals on the same representation used for classification, we expect the proven power of the dense trajectories [27, 28] to benefit the proposal generation. For our Action localization Proposals from dense Trajectories (APT) we cast the strong points of current proposal methods in an efficient clustering algorithm inspired by trajectory clustering for video segmentation, such as the seminal work of Brox & Malik [2].

We have the following contributions. i) We base our proposals on the same dense trajectories used for classification; ii) we present an efficient clustering algorithm to deal with large numbers of trajectories; iii) we demonstrate that video segmentation is time consuming and that its goals do not necessarily align with the goals of localization proposals. Experiments show that APT is faster than current methods and outperforms the localization and classification accuracy of the state-of-the-art on the UCF sports, MSR-II and UCF101 datasets. In Fig 1 we illustrate our action localization proposals.

2 Related work

Several action localization methods apply an action classifier directly on the video. Examples include sliding 3D subvolume methods like spatio-temporal template matching [19], a 3D boosting cascade [10] and spatio-temporal deformable 3D parts [22]. Others maximize a temporal classification path of 2D boxes through static frames [7, 23, 24, 30, 33] or search for the optimal classification result with a branch and bound scheme [34]. The benefit is that these methods do not require an intermediate representation and directly apply a classifier to sampled parts of the video. The disadvantage, however, is that they have to repeat the same procedure for each individual action class separately which is impractical for larger numbers of action classes. Instead, we use *unsupervised* spatio-temporal proposals to first generate a small set of bounding-box tubes that are likely to contain any type of action.

Current spatio-temporal proposals are inspired by 2D object proposals in static images. A version of objectness [1] is extended to video [26], selective search [25] led to tubelets from motion [9] and randomized Prim [15] was generalized to a spatio-temporal variant [17]. Several 2D object proposal methods and their 3D generalizations are based on a super-pixel segmentation pre-processing step [1, 9, 12, 15, 17, 25, 26] which we argue is too complex a problem for proposal generation. The use of segmentation makes proposal methods computationally too demanding for large scale video processing. To avoid the complex pre-processing step altogether, we propose a method of generating proposals without any pre-processing. We generate proposals from local dense trajectory features. These exact trajectory features are re-used later on for building a representation for action classification.

Local features provide a solid base for video segmentation [2], action recognition [13, 27] and action localization [9, 33]. Points are sampled at salient spatio-temporal locations [6, 13], densely [20, 31] or along dense trajectories [27, 28]. The points are represented by powerful local descriptors [4, 11, 14] that are robust to modest changes in motion and in appearance. Robustness to camera motion is either directly modeled from the video [8, 27] or dealt with at the feature level by the MBH descriptor [4, 28]. After aggregating local descriptors in a global video representation such as VLAD [8] or Fisher [16, 18, 27] they are input to a classifier like SVM. Due to the excellent performance of dense trajectory features [27, 28] in action localization [9, 33], we adopt them as our feature representation.

3 Action Proposals from dense Trajectories

3.1 Can we use current approaches?

It may be possible to use current spatio-temporal action proposal methods on the dense trajectories. We analyze the tubelets from Jain et al. [9] and prim3D from Oneata et al. [17]. The first step of both methods is a video segmentation, assigning a single segment ID to each pixel in the video. The second step is grouping of segments into action-proposals. Both algorithms compute feature similarities between all segment pairs that have neighboring spatio-temporal pixels. The tubelets use a greedy hierarchical clustering approach: merge the pair of most similar segments into a single segment; compute the features of this new segment and the pairwise similarities with its spatio-temporal neighbors; repeat till convergence. The prim3D approach applies a classifier on each segment pair to compute the likelihood of a good merge and then uses a randomized hierarchical clustering method: stochastically sample a classifier score; add segments to the hierarchy; repeat till convergence. The third step is the actual classification of the spatio-temporal proposals. Dense trajectories are computed for the video and each spatio-temporal proposal encodes its features into a representation fed to an action classifier.

A strong point of tubelets is that the hierarchical clustering is unsupervised, which alleviates the need for any training data. An advantage of prim3D is that it does not need to recompute similarities after every merge. For n elements, the time complexity of prim3D is $O(n^2)$, whereas recomputing similarities has a worst-case time complexity of $O(n^3)$. Regarding space complexity, both tubelets and prim3D have a $O(n^2)$ worst case complexity. Even a time efficient implementation for prim3D in a max-heap [15], in the worst case, will have to store $n(n-1)/2 = O(n^2)$ similarities. Unfortunately, more realistic videos of over 30 seconds may easily contain three hundred thousand trajectories, which would require an impractical 180GB of memory. Hence, we propose a new method, building on the strong points of tubelets and prim3D.

Property	Oneata et al., ECCV 2014 [17]	Jain et al., CVPR 2014 [9]	APT (ours)
Unsupervised	✗	✓	✓
Segmentation free	✗	✗	✓
Time complexity	$O(n^2)$	$O(n^3)$	$O(n^2)$
Space complexity	$O(n^2)$	$O(n^2)$	$O(n)$

Table 1: **Properties of recent spatio-temporal proposal algorithms** [9, 17]. APT is designed to include the strong points of existing algorithms: unsupervised, no additional segmentation step, and efficient in time and space.

3.2 Proposals from trajectories with localized SLINK

Because existing spatio-temporal proposal methods are not suitable for dense trajectories, we need another method that can adopt the strongest points of tubelets [9] and prim3D [17]. We base our method on the SLINK algorithm [21] which iteratively merges the best pair of similarities in a single-link hierarchal clustering. SLINK is unsupervised, does not require recomputing similarities, has $O(n^2)$ time complexity and crucially only $O(n)$ space complexity. This adopts all benefits from tubelets and prim3D, while avoiding their impractical properties. We summarize the algorithmic properties of these methods in Table 1.

Data: distance d, trajectory neighbors NN
Result: clusters in pointer representation $\{\Pi, \Lambda\}$
$\Pi[1] = 1; \Lambda[1..N] = \infty; M[1..N] = \infty$;
for $i \leftarrow 2$ **to** N **do**
  $\Pi[i] = i; \Lambda[i] = \infty; c = 1;$
  ids = NN[i]; M[ids]=d[i,ids];
  while $c \leq |ids|$ **do**
    $j = \text{ids}[c], q = \Pi[j];$
    if $\Lambda[j] \geq M[j]$ **then**
      $M[\Pi[j]] = \min(M[\Pi[j]], \Lambda[j]);$
      $\Lambda[j] = M[j]; \Pi[j] = i;$
    else
      $M[\Pi[j]] = \min(M[\Pi[j]], M[j]);$
    end
    if $q \notin ids$ **then**
      insert q in sorted ids;
    end
    $M[j] = \infty; c = c + 1;$
  end
  for $j \leftarrow 1$ **to** i **do**
    if $\Lambda[j] \geq \Lambda[\Pi[j]]$ **then**
      $\Pi[j] = i;$
    end
  end
end

Algorithm 1: Localized Slink.

The similarities that seed the proposal generation are based on improved dense trajectory features [27]. The features cover appearance with HOG, trajectory shape with TRAJ, motion with HOF, and change in motion with MBH. In addition, we also use the spatio-temporal position SPAT. A combination of multiple features by multiplying similarity scores allows capturing action type variation [9, 17], where more variation improves recall. We experimentally evaluate individual features and their normalizations.

The SLINK algorithm uses a 'pointer representation' to index clusters, see [21] for details. The elegance of this representation is that it can recursively be updated, allowing a single pass through all neighboring similarities. The pointer representations consists of two arrays Π and Λ of size n, where n is the number of trajectories. A cluster is represented by its member with the highest index i, $\Pi[i]$ is then the cluster (i.e. highest index) that point i joins, and $\Lambda[i]$ is the similarity between i and cluster $\Pi[i]$ when they join. The pointer representation describes a hierarchical clustering tree with n clusters. We cut the tree to output proposals when the difference between two merging clusters is larger than 50 trajectories. This value was found in practice to give a fair balance between the size of a cluster and the number of clusters.

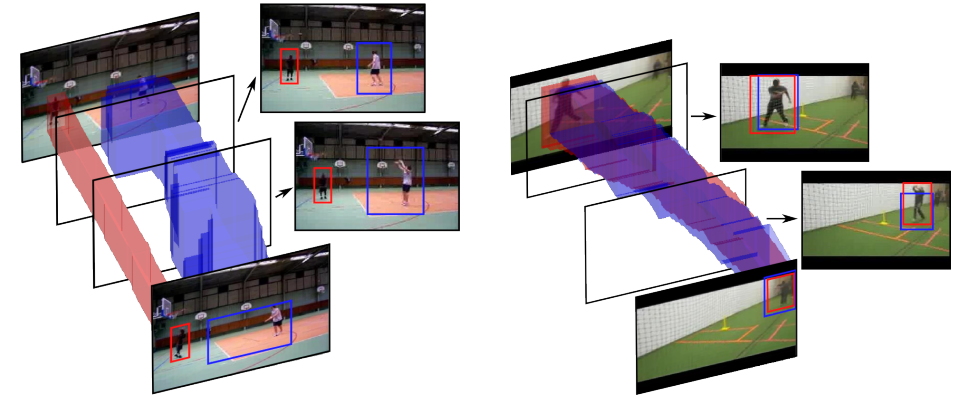

Figure 2: **Success and failure case of APT**. In the left video our method (blue tube) ignores the standing person (red tube) and tracks the moving actor. In the right video, there is ample variation in depth and position yet APT tracks the action well. Our method is intended for actions and thus works best when motion is present. See Fig 1 for another successful action localization result.

To enforce spatio-temporal compactness of the proposals we only consider the k spatio-temporal neighbors of each trajectory. We use efficient approximate k-best NN search with the Yael library [5]. The original SLINK algorithm, however, takes similarities between all points into account. We increase the efficiency of the original SLINK algorithm by using a sorted list of point neighbors, which we can search in $O(log(k))$. Note that the value k in the nearest neighbor search is a lower bound on the number of neighbors per point since the best k neighbors is not a symmetric relation. We will experimentally evaluate various values of k, and the relation between the temporal axis and the spatial axis for the spatio-temporal neighbors. Pseudo code for our modified version of SLINK is given in Algorithm 1 and we will make python code available.

4 Experiments

4.1 Datasets and evaluation

We evaluate on three diverse datasets for action localization: **UCF Sports, UCF 101** and **MSR-II**. UCF Sports consists of 150 videos extracted from sports broadcasts of varying resolution; it is trimmed to contain a single action in all frames. UCF101 is collected from YouTube and has 101 action categories where 24 of them contain localization annotations, corresponding to 3,204 videos. All UCF101 videos contain exactly one action, most of them (74.6%) are trimmed to fit the action. In contrast, the MSR-II Action dataset consists of 203 actions in 54 videos where each video has multiple actions of three classes. The actions are performed in a crowded environment with ample background movement. The MSR-II videos are relatively long, on average 763 frames, and the videos are untrimmed. Thus actions are temporary unconstrained which adds temporal localization. Compared to the trimmed UCF-Sports/UCF101 videos, the untrimmed nature of MSR-II makes this dataset the most realistic and the most challenging.

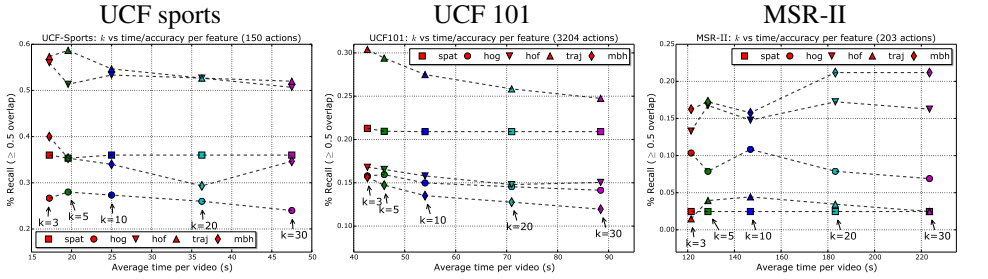

Figure 3: **Evaluating APT properties**. Analysis of computation time (x-axis) versus recall (y-axis) for the five base features: SPAT, HOG, HOF, TRAJ, MBH, for various values of k. Similar *marker color* denote a similar value of k, while the *marker shape* connected with dashed lines represent a single feature. Note that MBH does not perform well on trimmed UCF-Sports/UCF 101, while MBH is the best for untrimmed MSR-II. We draw 2 conclusions: i) in practice, computation time scales linearly in k; ii) No single feature will lead to good performance, heterogeneous features are key.

The quality of a proposal P with a ground truth tube G is evaluated with spatio-temporal tube overlap measured as the average "intersection-over-union" (IoU) score for 2D boxes for all frames where there is either a ground truth box or a proposal box. More formally, for a video V of F frames, a tube of bounding boxes is given by $(B_1, B_2, ...B_F)$, where $B_f = \emptyset$, if there is no action i in frame f. The IoU score between G and P is

$$\text{IoU}(G,P) = \frac{1}{|\phi|} \sum_{f \in \phi} \frac{G_f \cap P_f}{G_f \cup P_f}, \tag{1}$$

where ϕ is the set of frames where at least one of P_f, G_f is not empty. A detection is correct if the action is correctly predicted and the localization score is accurate, i.e., $\text{IoU}(G,P) \geq \sigma$. We set the threshold for the localization score to $\sigma = 0.5$ to evaluate recall, as commonly done for object detection [1, 15, 25]. When comparing with other action localization methods, we also report on lower σ values if $\sigma \geq 0.5$ is not reported.

4.2 Evaluating APT properties

The effect of the k-NN parameter. The k-parameter sets the number of spatio-temporal neighbors considered in the grouping. In Fig 3 we illustrate its effect on the computation time and the recall accuracy. For practical values of $k \leq 30$ the computation time is linearly related to k. The recall accuracy for each single feature is not heavily influenced by the value of k, from now on we use $k = 10$.

Which trajectory features to use. We plot recall accuracy for each of the five base features: SPAT, HOG, HOF, TRAJ, MBH independently in Fig 3. Note the role switch of the TRAJ and MBH features for trimmed UCF-Sports/UCF101 versus untrimmed MSR-II. The TRAJ feature performs well on the trimmed UCF-Sports and UCF101, while TRAJ scores badly on MSR-II. For MSR-II, the best single feature is MBH which performs sub-par on UCF101/UCF-Sports. The untrimmed videos of MSR-II benefit from motion boundaries for temporal segmentation while the majority of trimmed videos in UCF101 can focus on spatial trajectories since temporal segmentation is not an issue in trimmed videos. We conclude

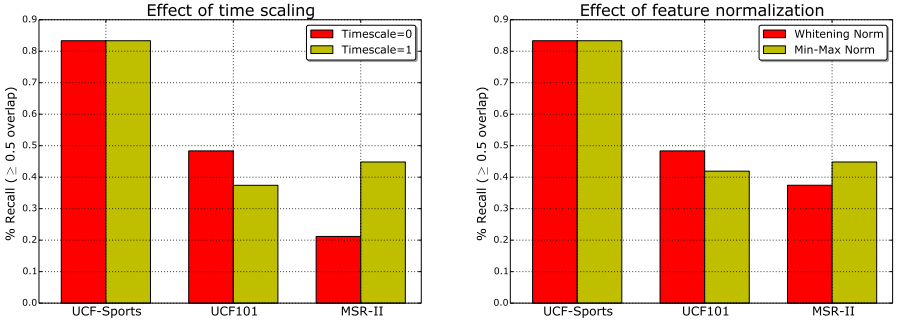

Figure 4: **Evaluating APT properties**. Left: Effect of scaling the temporal-axis in APT. Right: Effect of feature normalization when combining features. Note the conflict for trimmed (UCF101) vs untrimmed (MSR-II) in both graphs.

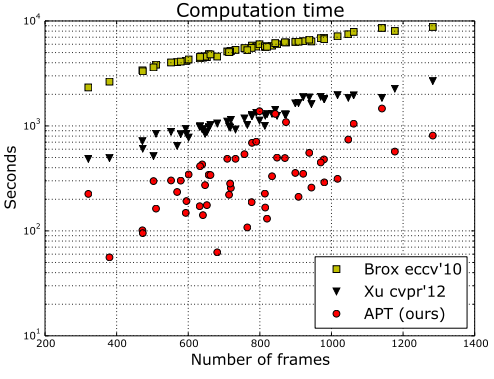

Figure 5: **Evaluating APT computation time** on all videos of MSR-II. Note the log scale on the y-axis. APT is typically faster than the pre-processing used for video segmentation (Xu & Corso [32]) in the tubelets localisation proposals [9]. APT is an order of magnitude faster than the trajectory clustering of Brox & Malik [2].

that no single feature is best, and heterogeneous features are essential. We use all feature combinations for generating proposals.

The importance of the time scale. For trimmed videos the temporal axis is not important and we evaluate setting the time scale to 0 when computing the kNN. The results in Fig 4 (left) show no effect for UCF-Sports. For UCF101 results improves significantly by ignoring time, whereas for MSR-II ignoring the time scale reduces performance dramatically. This shows that temporal modeling is important for untrimmed videos, whereas time should be ignored for trimmed videos. For APT we ignore the time scale for UCF-Sports and UCF101.

Feature normalization. The dense trajectory features ranges vary. Thus when similarities are combined by multiplication, their weight is not equal. We evaluate two feature normalization schemes: i) min-max scaling of the similarities and ii) whitening the similarities in a video. The results in Fig 4 (right) show that UCF-Sports is agnostic to normalization. UCF101 has a preference for whitening, whereas MSR-II has a preference for min-max scaling. This illustrates another difference between trimmed and untrimmed videos, and we use whitening for UCF-Sports, UCF101 and min-max for MSR-II.

Success and failure cases. In Fig 1 and Fig 2, we show success and failure cases of APT. When static actions are performed, our method has difficulties finding the action. On the other hand, APT performs well even with severe camera-motion as in Fig 1 and large scale changes as in Fig 2. Since APT is based on dense trajectories, which are meant for action recognition, our proposals perform best when motion is present.

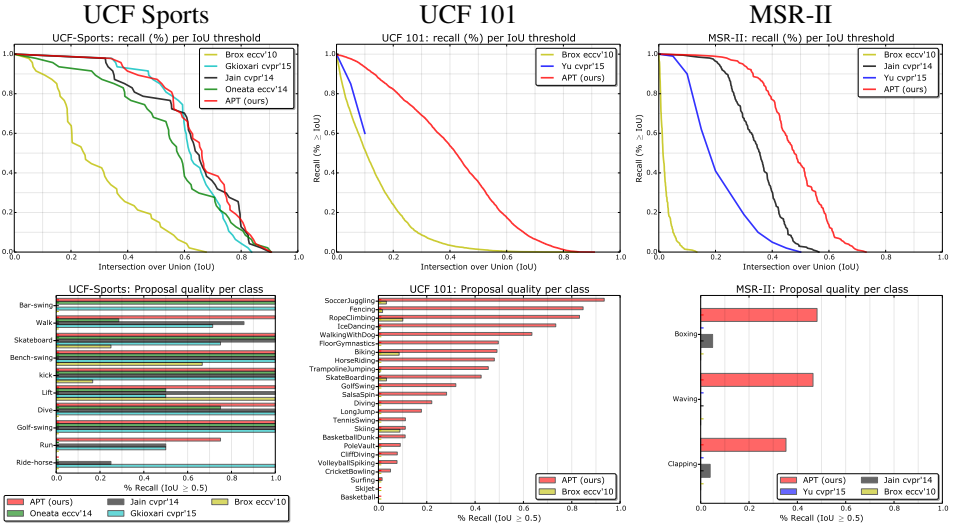

Figure 6: **Comparing action proposals**. Top row: recall for a varying threshold σ. Bottom row: recall per action category, where not all categories have proposals for $\sigma = 0.5$. For UCF Sports, exception for *Ride-Horse*, we are competitive with others, and even outperforming all others for *Run*. For UCF101 and MSR-II we outperform others on all actions.

4.3 Comparing action proposals

Speed. We evaluate computation speed on MSR-II which contains longer videos. Longer videos have relatively less overhead and thus provide more accurate measurements than short videos. In Fig 5 we compare the speed of our proposals against flow clustering of Brox & Malik [2] and the GB segmentation method by Xu & Corso [32]. GB is used as pre-processing in the spatio-temporal proposals by Jain et al. [9]. APT is typically faster than pre-processing [32] and an order of magnitude faster than flow clustering [2].

Proposal quality. In Fig 6 we show the proposal quality for the three datasets, including an analysis of recall per action category.

We first compare our method against an alternative trajectory clustering by Brox & Malik [2] on all three datasets. We used the author recommended settings, and fit tubes around the output segment. As can be seen in Fig 6 this method [2] is not well-suited for action proposals as it offers a precise, high-quality object segmentation. The goals for object proposals, however, are different: 1. a loose bounding box tube instead of a precise segmentation; 2. multiple hypotheses instead of a single grouping. We conclude that the goals of video segmentation do not necessarily align well with the goals for action proposals. We detail a comparison with the state-of-the-art per dataset, next.

On UCF Sports we perform best for 9 out of 10 actions. The only exception is *Ride-Horse*, where only the fully *supervised* method of Gkioxari & Malik [7] gives excellent results, suggesting that this class is hard for unsupervised methods based on motion. For all other actions our proposals are competitive or improve over the state-of-the-art [7, 9, 17]. For the motion-rich category *Run* we outperform all other methods, illustrating the power of dense trajectories to capture motion proposals. For UCF101 and MSR-II we outperform others on all categories. For UCF 101 we compare against the recent work of Yu & Yuan [33] who only report recall for 0.1 and 0.05 IoU. We outperform this method with a large margin

	ABO	MABO	Recall	#Proposals
UCF Sports				
Brox & Malik, ECCV 2010 [9]	29.84	30.90	17.02	**4**
Jain et al., CVPR 2014 [9]	63.41	62.71	78.72	1,642
Oneata et al., ECCV 2014 [24]	56.49	55.58	68.09	3,000
Gkioxari & Malik, CVPR 2015 [9]	63.07	62.09	87.23	100
APT (ours)	**65.73**	**64.21**	**89.36**	1,449
UCF 101				
Brox & Malik, ECCV 2010 [9]	13.16	12.68	1.40	**3**
APT (ours)	**42.43**	**41.90**	**36.84**	2,299
MSR-II				
Brox & Malik, ECCV 2010 [9]	2.28	2.34	0	**6**
Jain et al., CVPR 2014 [9]	34.88	34.81	2.96	4,218
Yu & Yuan, CVPR 2015 [33]	n.a.	n.a.	0	37
APT (ours)	**48.02**	**47.87**	**44.33**	6,706

Table 2: **Comparing action proposals** for commonly used metrics against other methods, where *n.a.* denotes not reported values. *ABO* is the Average Best Overlap, *MABO* the Mean ABO over all classes, *Recall* is measured at an IoU overlap $\sigma \geq 0.5$ and the number of proposals is averaged per video. APT outperforms others with a modest number of proposals.

and also report localization results for IoU ratios up to 1.0. The category-recall is not reported in [33], thus we do not show per-category results for this method. For the untrimmed MSR-II dataset we compare against [9, 33]. For more precise localization results on MSR-II with IoU overlap scores of $\sigma \geq 0.2$ we outperform both methods with a large margin (top right, Fig 6). The results per category re-iterate this point, where [33] reports no proposals above the IoU overlap threshold of $\sigma \geq 0.5$.

In Table 2 we summarize proposal results with commonly used metrics. With a modest number of proposals our method outperforms all others on ABO, MABO and Recall.

4.4 Classifying action proposals

For classification we re-use the improved dense trajectories [27], exploiting exactly the same trajectory features as used for creating the action proposals. We compute a standard Fisher vector [16, 18, 27] aggregated over the trajectories that form a proposal. We concatenate all descriptors to a 426d vector, use PCA to half the dimensionality to 213d and fit 128 GMMs. For training, we use a one-vs-rest SVM classifier with a Hellinger kernel.

In Fig 7 (left) we show the Area Under the ROC curve for a varying recall threshold σ on UCF-Sports. The fully *supervised* method of Gkioxari & Malik [9] gives excellent results. Their method, however, does not use spatio-temporal proposals, and needs to temporally aggregate detection results for each class separately. This scales poorly to larger datasets with more classes. Our results are competitive with the state of the art, while retaining fully unsupervised action proposals.

For the large UCF 101 dataset we follow [33] and report mean average precision (mAP) over all 24 classes. In Fig 7 (middle) we show mAP scores for increasing IoU threshold qualities and compare with the recent work of Yu & Yuan [33] that evaluated on this dataset. They report mAP for only a single IoU threshold, which we outperform.

For MSR-II we follow standard practice and train on KTH [3, 33]. In Fig 7 (right) we show mAP scores for a varying IoU threshold. To reduce clutter we only show the best reported number on this dataset [33]. A comparison with the full state of the art is given in Table 3, where an IoU threshold of 0.125 is common. We are best for *Clap*, and second for *Box*. On average we outperform all methods on MSR-II.

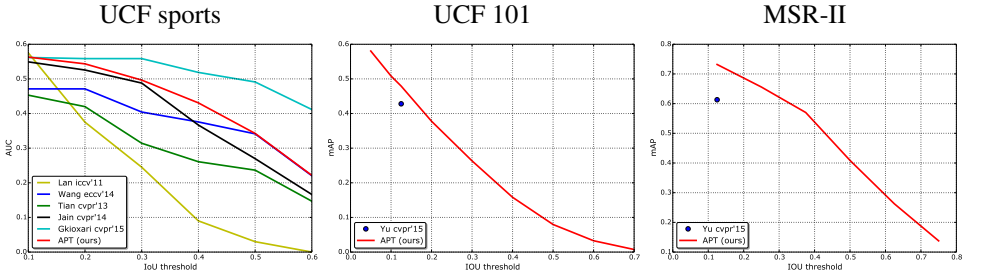

Figure 7: **Classifying action proposals** with a varying IoU threshold σ. Left: AUC on UCF Sports, the fully supervised method of Gkioxari & Malik [7] is best, we are competitive with the unsupervised state of the art. Middle: mAP for UCF 101, we outperform the recent scores on this dataset by Yu & Yuan [33]. Right: mAP for MSR-II, where we outperform the best results on this set by Yu & Yuan [33]. To reduce clutter for MSR-II we only plot the best results, others are in Table 3.

Table 3: **Classifying action proposals** on MSR-II for a threshold of 0.125. Our average precision score is best for *Clap* and runner-up for *Boxing*. On average we outperform all other methods.

Method	Boxing	Clap	Wave	mAP
Cao et al., CVPR 2010 [3]	17.48	13.16	26.71	19.12
Tian et al., CVPR 2013 [22]	38.86	23.91	44.70	35.82
Jain et al., CVPR 2014 [9]	46.00	31.41	**85.79**	54.40
Yuan et al., TPAMI 2011 [34]	64.90	43.10	64.90	55.30
Wang et al., ECCV 2014 [29]	41.70	50.20	80.90	57.60
Yu & Yuan, CVPR 2015 [33]	**67.40**	46.60	69.90	61.30
APT (ours)	67.02	**78.40**	74.14	**73.19**

5 Conclusion

We describe APT: an apt method for spatio-temporal proposals for action localization based on the same dense trajectories as proved effective as a video representation for action classification. By reusing the trajectories, we preempt any pre-processing such as video segmentation as used by other proposal methods. Our method is efficient and outperforms the state of the art on localization and classification on three datasets. Results show that APT is particularly effective for realistic untrimmed videos where temporal localization is essential. These results are promising to facilitate action localization for large realistic video collections.

6 Acknowledgement

This research is supported by the STW STORY project and the Dutch national program COMMIT.

References

[1] B. Alexe, T. Deselaers, and V. Ferrari. Measuring the objectness of image windows. *TPAMI*, 34(11), 2012.

[2] T. Brox and J. Malik. Object segmentation by long term analysis of point trajectories. In *ECCV*, 2010.

[3] L. Cao, Z. Liu, and T. Huang. Cross-dataset action detection. In *CVPR*, 2010.

[4] N. Dalal, B. Triggs, and C. Schmid. Human detection using oriented histograms of flow and appearance. In *ECCV*, 2006.

[5] M. Douze and H. Jégou. The yael library. In *MM*, 2014.

[6] I. Everts, J. van Gemert, and T. Gevers. Evaluation of color spatio-temporal interest points for human action recognition. *TIP*, 23(4):1569–1580, 2014.

[7] G. Gkioxari and J. Malik. Finding action tubes. In *CVPR*, 2015.

[8] M. Jain, H. Jégou, and P. Bouthemy. Better exploiting motion for better action recognition. In *CVPR*, 2013.

[9] M. Jain, J. van Gemert, H. Jégou, P. Bouthemy, and C. Snoek. Action localization with tubelets from motion. In *CVPR*, 2014.

[10] Y. Ke, R. Sukthankar, and M. Hebert. Efficient visual event detection using volumetric features. In *ICCV*, 2011.

[11] A. Kläser, M. Marszalek, and C. Schmid. A spatio-temporal descriptor based on 3d-gradients. In *BMVC*, 2008.

[12] P. Krähenbühl and V. Koltun. Geodesic object proposals. In *ECCV*, 2014.

[13] I. Laptev and T. Lindeberg. Space-time interest points. In *ICCV*, 2003.

[14] I. Laptev, M. Marzalek, C. Schmid, and B. Rozenfeld. Learning realistic human actions from movies. In *CVPR*, 2008.

[15] S. Manen, M. Guillaumin, and L. van Gool. Prime object proposals with randomized prim's algorithm. In *ICCV*, 2013.

[16] D. Oneata, J. Verbeek, and C. Schmid. Action and Event Recognition with Fisher Vectors on a Compact Feature Set. In *ICCV*, 2013.

[17] D. Oneata, J. Revaud, J. Verbeek, and C Schmid. Spatio-temporal object detection proposals. In *ECCV*, 2014.

[18] X. Peng, C. Zou, Y. Qiao, and Q. Peng. Action recognition with stacked fisher vectors. In *ECCV*, 2014.

[19] M. Rodriguez, J. Ahmed, and M. Shah. Action mach: a spatio-temporal maximum average correlation height filter for action recognition. In *CVPR*, 2008.

[20] F. Shi, E. Petriu, and R. Laganiere. Sampling strategies for real-time action recognition. In *CVPR*, 2013.

[21] R. Sibson. Slink: an optimally efficient algorithm for the single-link cluster method. *The Computer Journal*, 16(1):30–34, 1973.

[22] Y. Tian, R. Sukthankar, and M. Shah. Spatiotemporal deformable part models for action detection. In *CVPR*, 2013.

[23] D. Tran and J. Yuan. Max-margin structured output regression for spatio-temporal action localization. In *NIPS*, 2012.

[24] D. Tran, J. Yuan, and D. Forsyth. Video event detection: From subvolume localization to spatio-temporal path search. *TPAMI*, 36(2):404–416, 2014.

[25] J. Uijlings, K. van de Sande, T. Gevers, and A. Smeulders. Selective search for object recognition. *IJCV*, 104(2):154–171, 2013.

[26] M. Van Den Bergh, G. Roig, X. Boix, S. Manen, and L. van Gool. Online video seeds for temporal window objectness. In *ICCV*, 2013.

[27] H. Wang and C. Schmid. Action Recognition with Improved Trajectories. In *ICCV*, 2013.

[28] H. Wang, A. Kläser, C. Schmid, and C. Liu. Action recognition by dense trajectories. In *CVPR*, 2011.

[29] L. Wang, Y. Qiao, and X. Tang. Video action detection with relational dynamic-poselets. In *ECCV 2014*, 2014.

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

[31] G. Willems, T. Tuytelaars, and L. van Gool. An efficient dense and scale-invariant spatio-temporal interest point detector. In *ECCV*, 2008.

[32] C. Xu and J. Corso. Evaluation of super-voxel methods for early video processing. In *CVPR*, 2012.

[33] G. Yu and J. Yuan. Fast action proposals for human action detection and search. In *CVPR*, 2015.

[34] J. Yuan, Z. Liu, and Y. Wu. Discriminative video pattern search for efficient action detection. *TPAMI*, 33(9), 2011.