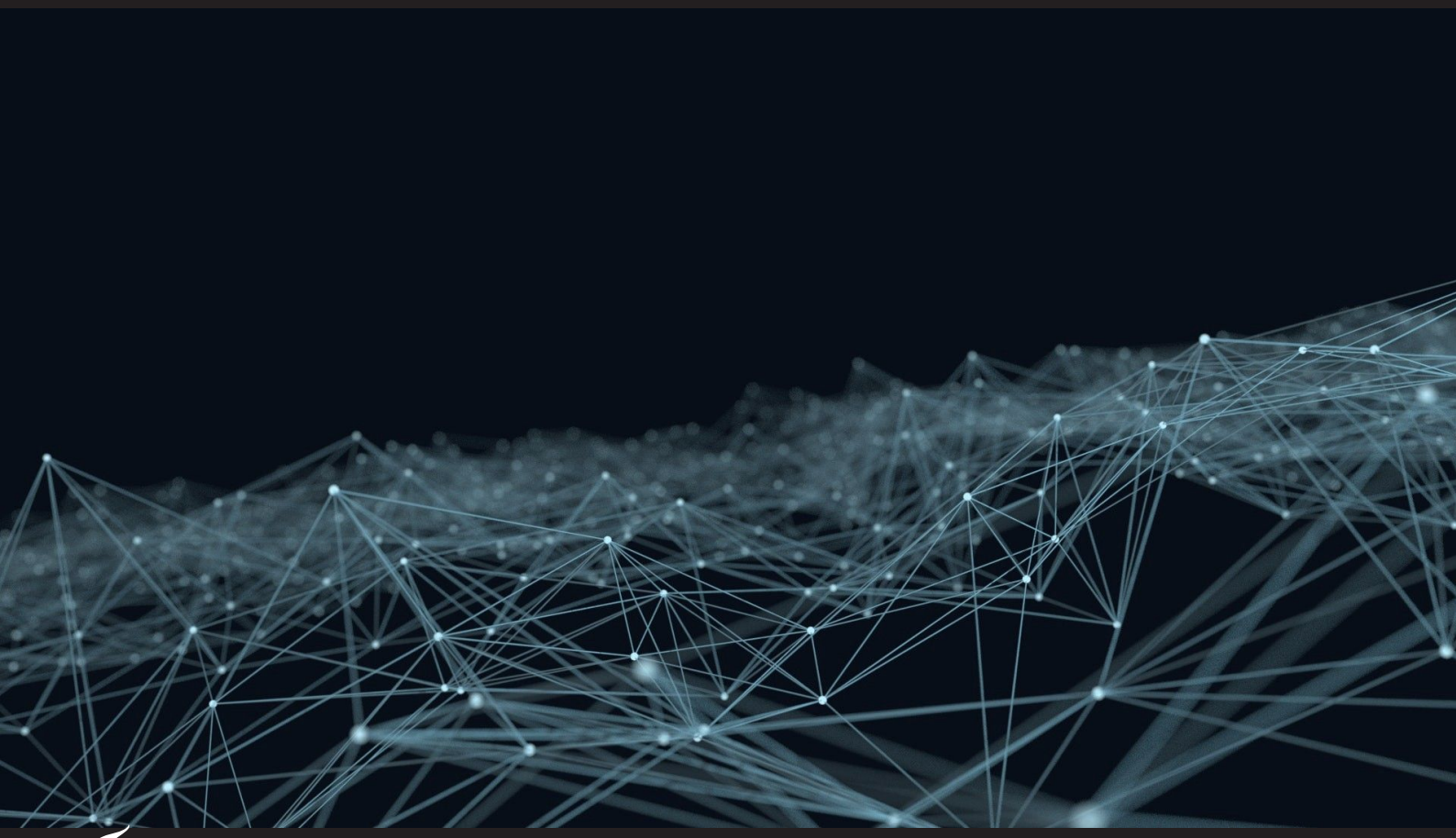
PUNet

Temporal Action Proposal Generation with Positive Unlabeled Learning using Key Frame Annotations

Noor ul Sehr Zia



TU Delft

PUNet

Temporal Action Proposal Generation with Positive Unlabeled Learning using Key Frame Annotations

by

Noor ul Sehr Zia

in partial fulfillment of the requirements for the degree of

Master of Science in Computer Science
Track: Data Science & Technology
(Pattern Recognition & Bioinformatics)

at Delft University of Technology,
Faculty of Electrical Engineering, Mathematics and Computer Science,
to be defended publicly on Monday August 31, 2020.

Student number:	4822498	
Project duration:	December 1, 2020 – August 31, 2020	
Thesis committee:	Dr. Jan van Gemert,	TU Delft, Supervisor, Committee Chair
	Dr. Silvia-Laura Pintea,	TU Delft, Committee Member
	Dr. Ujwal Gadiraju,	TU Delft, Committee Member

TU Delft

Preface

This report presents the work of my master's thesis project on the topic of Temporal action proposal generation using key frame annotations. This research was conducted at Computer Vision Lab of Pattern Recognition and Bioinformatics Group in TU Delft under the supervision of Dr. Jan Van Gemert.

First and foremost, I would like to express my deepest appreciation for my supervisor Dr. J.C. van Gemert for guiding me through the thesis process. I would like to express my gratitude to Osman Kayhan for being a mentor during my thesis duration and providing constant encouragement and guidance. I would also like to thank Dr. Silvia Pintea and Dr. Ujwal Gadiraju for their interest in my thesis and for evaluating my work.

I am deeply grateful to my family and friends for their continuous love and support. Their constant motivation has allowed me to focus on my work with a clear mindset.

Noor ul Sehr Zia
Delft, August 2020

Contents

1

Scientific Paper

PUNet: Temporal Action Proposal Generation with Positive Unlabeled Learning using Key Frame Annotations

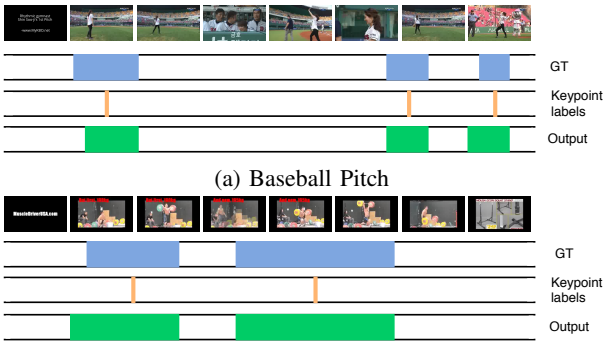
Noor ul Sehr Zia
Delft University of Technology
N.U.S.Zia@student.tudelft.nl

Osman Semih Kayhan
Delft University of Technology
O.S.Kayhan@tudelft.nl

Jan C. van Gemert
Delft University of Technology
J.C.vanGemert@tudelft.nl

***Abstract*—A good action proposal method should generate proposals with high recall and high temporal overlap with groundtruth. The quality of the proposals relies on the labeled data available during training. Obtaining labeled data for untrimmed videos is a time consuming, expensive and error-prone task. The labels obtained are also subjective and the temporal bounds are inconsistent between different human annotators. We propose using a single key frame label for each action instance instead of the start and end point labels to generate temporal proposals. This reduces the number of labeled action frames in the dataset leading to class imbalance. To overcome this, we replace the learning setting with a PU-learning setup.**

We demonstrate that using key frames as labels give high quality proposals and yield results comparable to using full annotations while being faster to annotate as the exact temporal bounds no longer need to be annotated. We evaluate our method on THUMOS'14 and ActivityNet v1.2 dataset. Further experiments indicate that by combining existing action classifier on our proposals, our method is able to achieve high mean average precision (mAP) for action localization.



(a) Baseball Pitch

(b) Clean and Jerk

Fig. 1. Our proposed method. A single frame is labeled for each action instance instead of start and end points of the action duration. The detected results are shown for two classes of THUMOS'14 dataset. Using a single frame, the PU learning network is able to detect the action boundaries with low error.

I. Introduction

The amount of video data available is increasing exponentially, raising the need for reliable video analysis methods. Most real world videos are untrimmed in nature and contain multiple action sequences along with background making it necessary to have an efficient temporal action localization algorithm. Temporal action localization aims to identify the frames containing actions for each action instance and the action classes.

Current action localization methods [1]–[3] extend from object detection. Object proposals have significantly improved the object detection methods and have contributed to large scale detection in terms of efficiency and high detection rates [4], [5]. The current object detection pipelines are divided into two steps: proposal generation and object classification. Inspired from this, recent action detection methods first generate proposals and then perform classification [6], [7]. For untrimmed video data, the goal is to have a fast localization and using proposal methods can speed up the localization process. These proposal methods are fully supervised and require action temporal bounds to generate proposals.

The performance of action proposal generation networks is limited by the annotated data available. In untrimmed datasets, each action instance is labeled with a start and end timestamp of the action and each video can have multiple action instances which may occur at same timestamp [8]. Obtaining these labels is time consuming and expensive. Moreover, the labeling is subjective and error prone [9] due to different understanding of action duration, thus affecting the results of the model trained using these labels [10]. Recent works in action recognition have shown that the performance improves by using most discriminative portions of the video for training [11]. Similarly, work has been done to optimize the segment length and recognize human actions with less number of frames [12], [13]. Using a single timestamp instead of start and end time for action recognition has been shown to be a reasonable compromise between recognition performance and annotation effort [14]. A single point annotation is also significantly faster to obtain [15]. We use weakly supervised setup for action localization where instead of labeling start and end of the segment, we propose labeling a single action frame as "key frame" inside an action's temporal window. We choose the midpoint of the action instance as our key frame label. Instead of learning a sampling distribution [14] our method uses Positive Unlabeled (PU) learning to detect action frames.

We label one frame belonging to the action instance as our key frame. The remaining frames are now a combination of background and unlabeled action frames, referred together

as 'unlabeled data'. If we consider the unlabeled data as negative class, the problem becomes imbalanced due to the high ratio of unlabeled data to positive data and cannot be solved effectively using positive-negative learning. Therefore, the problem is translated to a PU learning [16] setting where the true positives are iteratively removed from the unlabeled data. In the beginning each unlabeled sample can be either:

- a background feature,
- a part of the action duration that is left unlabeled.

The main contributions of our work are:

1) We propose Positive Unlabeled Network (PUNet) for action proposal generation using a single labeled frame per action instance. We show that using key frame instead of start and end point gives similar performance when compared to fully supervised methods.
2) We demonstrate that our weakly supervised setting which utilizes less labels is able to achieve similar results to state of the art fully supervised settings.
3) Our method generalizes well and is able to generate proposals for unseen action classes.

II. Related Work

A. Temporal Action Localization

Temporal action localization determines where an action takes place in a video. It outputs the action boundaries and class. Recently, proposal based methods are being utilized for fully supervised action localization [1], [17]–[20]. Earlier works for proposal generation use sliding windows directly as proposals [21], [22]. But sliding windows lead to huge computational overhead for recognition due to redundant computations as overlapping frames are processed more than once. To avoid window overlap, recent techniques such as SST [23] use a GRU-based sequence encoder that is able to generate proposals in a single stream without dividing into overlapping windows. TAG [18] uses watershed algorithm to generate flexible proposals from actionness probabilities. BSN [17] determines local information as starting, ending and actionness probabilities of temporal locations and combines them with high probability locations, and evaluates proposal level features to generate proposals in "local to global" fashion. Most recent methods use matching networks to retrieve temporal proposals [24]. In our approach, we reduce the labels needed for proposal generation by only using a key frame label for training.

Recently weakly supervised techniques have been proposed for action localization that only utilize the video-level action labels to learn the temporal bounds and action labels from untrimmed videos. UntrimmedNets [25] use a hard and soft selection scheme to localize segments from classification scores. Hide and Seek approach [26] randomly hides frames in a video to make the network focus on most discriminative parts. In addition to these, BasNet [27] and BUME [28] explicitly model the background class as a separate network branch. These methods achieve good results at the expense of complex network architectures requiring large number of parameters and longer training time. We try to find a middle ground by using key frame labels with a one layer neural network in a PU learning setting.

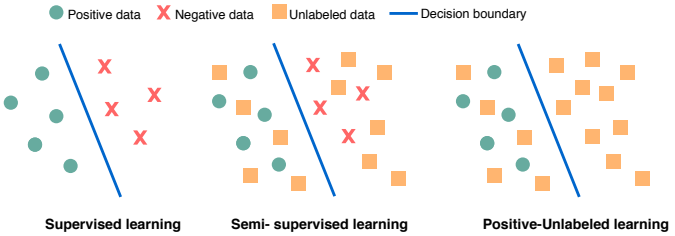


Fig. 2. Different types of learning settings. We use the positive unlabeled setting where few positive samples are labeled and remaining samples are used to extract reliable negatives.

B. Positive Unlabeled Learning

Data annotation is an expensive and laborious task. To reduce annotation effort, in recent years there has been an increased interest in developing algorithms that do not require fully annotated data. The most common learning set up is supervised learning where the algorithm has labeled positive and negative samples. Semi supervised learning relaxes the requirement of labeling and the training data consists of some positive and negative labeled samples and rest of the data is unlabeled. An extension of semi supervised learning is PU learning where only few positive labeled samples are required and remaining data is considered to be unlabeled [16]. The unlabeled data can consist of both positive and negative samples. We illustrate these differences in Figure 2. The original method of PU learning by Liu et al. [29], [30] isolates a set of reliable negatives from the unknown data and increases the set until no new negatives are identified. Fusilier et al. [31] propose a variant of the original method that isolates a set of reliable negatives from unknown but iteratively refines the set. In this paper, we show that PU learning can be used to detect action frames from limited number of labeled groundtruth frames during training.

III. Method

A. Problem definition

An untrimmed video sequence can be denoted as $X = \{x_n\}_{n=1}^{T}$ with T frames where x_n is the n-th frame in the video. In this work, the annotations of untrimmed videos can be defined as $\Psi_g = \{\varphi_n = (t_{m,n})\}_{n=1}^{N_g}$ where $t_{m,n}$ is the midpoint of the action instance n which we refer to as our key frame and N_g is the total number of action instances. For proposal generation, the class labels are not considered and only the key frames are used.

B. Input

We use I3D pretrained on Kinetics dataset to extract RGB and optical flow (OF) features from a given video. The feature representations from RGB and OF are concatenated along the $y - axis$ to obtain $(T * 2D)$ features for a video of duration T. For each video, we use a frame from the action region

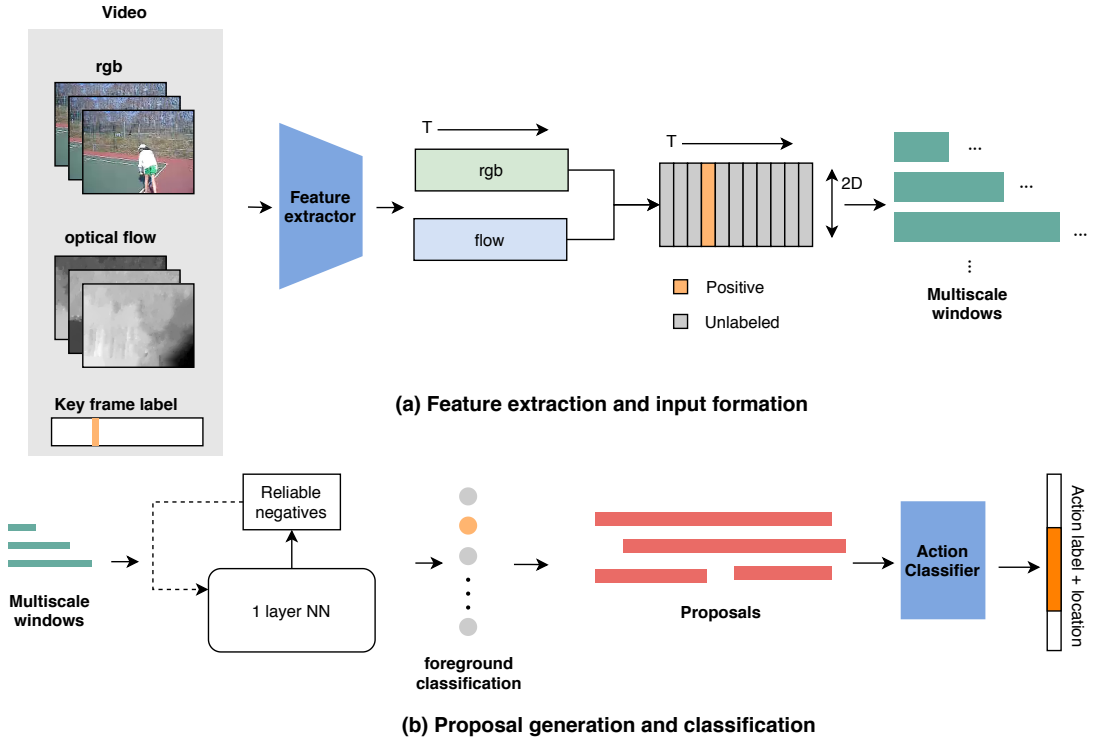


Fig. 3. Overview of our proposed approach. (a) An I3D network is used for encoding features and one point is labeled for each action instance. We divide the input into non-overlapping windows to be used for training. (b) The model is trained using PU learning technique at different scales to extract proposals. The proposals are classified using state of the art action classifier [25].

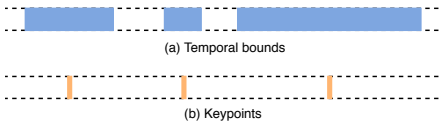


Fig. 4. For a given set of action instances, the key frame for that instance is the midpoint of the duration of the action. One key frame is selected irrespective of the duration of action instance.

as a key frame label. During training, only this single frame location is used as a label with a weakly-supervised manner. The key frames corresponding to groundtruth annotations can be seen in Figure 4. Regardless of the action duration, only a single key frame is used as a label.

Note that, each input feature corresponds to a binary label in terms of the key frame label and none of the action class labels are used for training or evaluation.

C. Multi-scale window generation

From untrimmed videos, we extract temporal windows of varied lengths 16, 32, 48, 64 and 80 frames with no overlap. Window label is positive if a key frame is present within that window duration.

D. PU learning

The modified PU-learning algorithm [31] is used to train the binary classifier. The algorithm finds negative samples that are most dissimilar from the positive samples by refining the 'reliable negatives'. Positive versus Unlabeled classifier is trained and tested on the unlabeled training set. The predicted negative samples with a high confidence score are considered to be reliable negatives. The remaining unlabeled samples are removed from training. The size of the reliable negatives is reduced iteratively by training a classifier using positive and reliable negative data and evaluating on reliable negative data points. Reliable negatives classified as positives are removed from the training set and this step is repeated until no positive classes are identified or the size of reliable negatives is less than positive samples. This reduces the size of the negative samples and overcomes class imbalance.

E. Proposal generation and classification

The proposal generation module uses PU classifier to generate candidate proposals for each window scale. The results from different window scales are combined to get the final proposals. We evaluate our proposals with a detection by classifying proposals approach and use state of the art action classifier [25]. The overview of PUNet can be seen in Figure 3.

IV. Experiments

A. Experimental Setup

1) Datasets: We evaluate PUNet on the widely used THUMOS'14 [32] and ActivityNet 1.2 [33] dataset. The THUMOS'14 dataset consists of temporal annotations for 20

TABLE I
Comparison of our method with other state of the art proposal generation methods on THUMOS'14 dataset in terms of AR@AN. Our method outperforms all fully supervised methods at AR@50 and AR@100 except BSN.

Supervision	Feature	Method	@50	@100
Full	C3D	DAPs [34]	13.56	23.83
	-	Sparse-prop [35]	13.42	21.44
	C3D	SCNN-prop [36]	17.22	26.17
	C3D	SST [23]	19.90	28.36
	C3D	TURN [37]	19.63	27.96
	flow	TURN [37]	21.86	31.89
	C3D	BSN [17]	29.58	37.38
	2-stream	BSN [17]	35.41	46.06
Weak - Key frames	I3D	PUNet	33.34	41.10

TABLE II
Comparison of our method with other state of the art proposal generation methods on ActivityNet v1.2 dataset in terms of AR@AN. Our method PUNet gives a higher recall while needing less number of proposals.

Supervision	Feature	Method	# proposals	AR@100
Full	C3D	DAP [34]	100	12.1
	STIP	TAP [35]	90	14.9
Weak	I3D	PUNet	20	20.3

classes comprising of 200 validation and 213 test videos. The validation videos are used for training. On average each video has 15 action instances. ActivityNet 1.2 consists of 100 action classes and 4819 training, 2383 validation and 2480 test videos. We use the validation videos for testing as the groundtruth for test videos are withheld.

2) Evaluation metric: PU-learning method is evaluated using F1-score. For temporal action proposal generation task, Average Recall (AR) calculated at different IoU thresholds is used for evaluation. We also calculate AR with average number of proposals (AR@AN) to determine relation between recall and number of proposals. For temporal action detection, mean average precision (mAP) is reported.

3) Implementation details: We use I3D pre-trained on Kinetics as our feature extractor. We use segments of 16 frames as input to our I3D network. The rgb and opti flow features are concatenated and the dimension of input feature x is 2048. We do not fine tune the feature extractor. For the classifier, we use a single layer Multi Layer Perceptron (MLP) with 100 hidden units. The network is trained using adam optimizer and 10^{-4} learning rate. To extract the initial set of reliable negatives, the predicted negatives are thresholded based on their confidence score. The threshold is set as 0.99.

B. Results

A good proposal generation method should generate high recall with less number of proposals. We determine our methods ability to do this by plotting the average recall against the average number of proposals for THUMOS'14 dataset (Figure 5a). PUNet outperforms most state of the art methods which use full supervision. We list the comparative results for THUMOS'14 and Activitynet v1.2 at AN = 50,100 in Table I and II. We also evaluate the quality of our generated proposals by comparing the recall at different tIoU thresholds (Figure 5b). Our results have significantly higher recall at 100 proposals for tIoU 0.1 to 0.5. The results for action detection indicate that PUNet performs better than few fully and weakly supervised methods. These results are presented in Table III and IV.

TABLE III
Comparison of our method with the state of the art methods on the THUMOS'14 dataset. Average mAP is reported. Weak * indicate use of additional information in weakly supervised approach. PUNet outperforms most weakly supervised action localization methods and some fully supervised methods while utilizing less annotations.

Supervision	Feature	Method	AVG mAP
Full	-	S-CNN [36]	19.9
	-	CDC [38]	22.8
	-	BSN [17]	36.8
Weak	I3D	STPN [39]	18.5
	UNT	AutoLoc [40]	21.0
	I3D	Liu et al. [41]	23.7
	I3D	BaS-Net [27]	27.3
	I3D	RPN [42]	27.6
	I3D	BUME [28]	30.0
*Weak **	I3D	PUNet	26.1

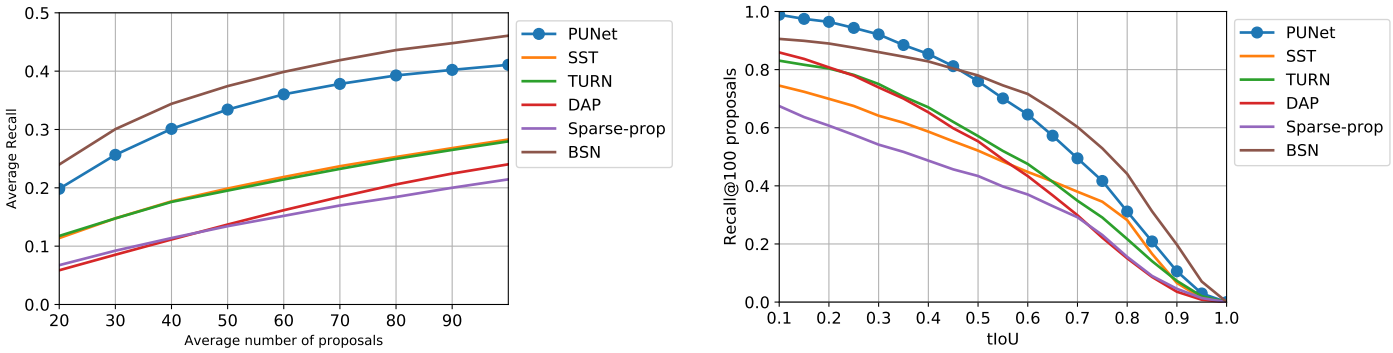
TABLE IV
Comparison on ActivityNet 1.2 with the current state of the art methods.The average mAP is reported. The results for I3D feature extractor are reported for techniques utilizing feature extraction. PUNet has comparable performance to fully supervised method and outperforms most weakly supervised methods for action localization.

Supervision	Method	AVG mAP
Full	S-CNN [36]	26.6
Weak	UntrimmedNets [25]	3.6
	AutoLoc [40]	16.0
	Liu et al. [41]	22.4
	W-TALC [43]	18.0
	BaS-Net [27]	24.3
	RPN [42]	23.3
	BUME [28]	25.4
*Weak **	PUNet	23.7

C. Ablation studies

We conduct some controlled experiments to test the contribution of each design decision.

1) Key frame position: We check different key frame positions as our network input. We test our network performance with midpoint, start, end and random point in the temporal boundary of action as our key frame label. We also check using multiple random points within the temporal window as our key frames. The results indicate random point gives 12.9% and 29% higher and midpoint gives 15.63% and 31.25% higher performance than start and end points respectively (Figure 6). Using start and end points as network input drastically reduce the performance. This is because the labels may not contain the precise temporal bounds and the exact temporal extent of an action is subjective [9]. The probability of a middle or random point inside the temporal duration of belonging to the true annotation window is higher than the start and end points, as the error margin is high on the endpoints. We also tested with three random points within the action duration as our labels,



(a) Average recall versus number of proposals

(b) Recall at tIoU thresholds for 100 proposals

Fig. 5. Comparison of our weakly supervised proposal generation method with the state of the art fully supervised methods on THUMOS'14 dataset. (a) PUNet is able to achieve high recall performance with few number of proposals. (b) Recall with 100 proposals at different tIoU thresholds show PUNet has high recall compared to all fully supervised methods when tIoU < 0.5. At higher tIoUs, PUNet outperforms all fully supervised methods except BSN.

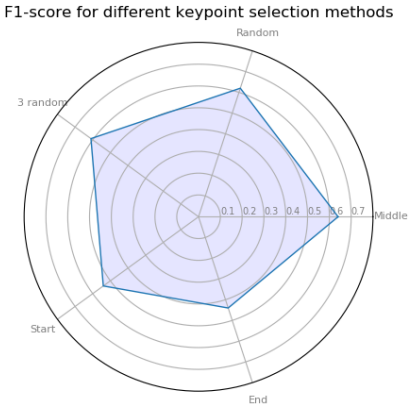


Fig. 6. Effect of different selection methods on model performance. Random and Middle point as labels give best results.

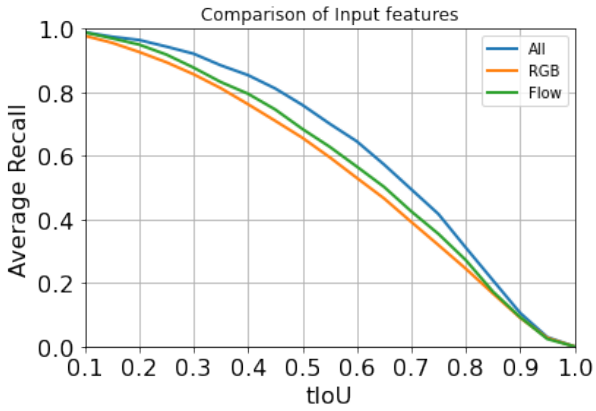


Fig. 7. Average Recall at different tIoU thresholds with RGB, optical flow and RGB+optical flow as network input. Combining optical flow and rgb features gives the best performance.

TABLE V
Ablation study of our method with different classifier and input type. F1-score is used to compare the settings. The results for SVM and MLP as base classifier and RGB, flow and RGB + flow as network input are compared. MLP as classifier and RGB+flow input give superior performance.

	Setting	**F1-score**
Classifier	SVM	0.64
	MLP	0.70
Input type	RGB	0.66
	Flow	0.67
	RGB + Flow	0.70

but did not notice any significant performance difference when compared with using a single random point.

2) Network: For the base classifier for PU-learning setup, two different classifiers are tested. We use a linear Support Vector Machine (SVM) and a single layer Multi Layer Perceptron (MLP) as our classifiers. The MLP network has higher F1-score on the test dataset (Table V).

3) Input data type: Most existing action analysis methods use RGB and optical flow as network input [17], [27], [28].

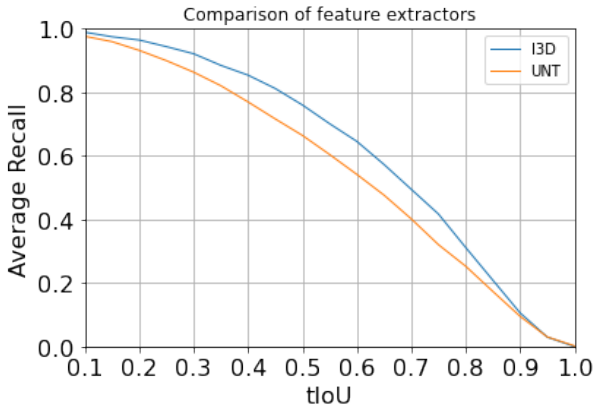


Fig. 8. Average Recall at different tIoU thresholds with different feature extractors. I3D achieves a superior performance compared to UntrimmedNets (UNT).

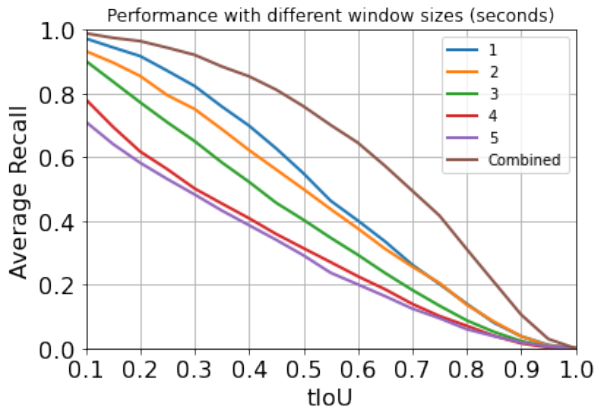


Fig. 9. Average Recall at different tIoU thresholds for different window sizes at 16 frames per second. Best results are achieved by combining the outputs from all window sizes.

We test PUNet with RGB, optical flow and concatenated RGB and optical flow feature inputs. Table V shows the results of the three feature representations in terms of background and foreground classification. The results of proposal quality in terms of recall at tIoU are shown in Figure 7.

4) Input features: We train PUNet with features extracted from UntrimmedNets [25] and I3D [44]. Figure 8 shows the proposal performance as recall vs tIoU curve. The I3D features outperform UntrimmedNets features. To notice that PUNet with untrimmed features outperforms most fully supervised techniques shown in Figure 5b. The performance of PUNet is not dependent on feature extractor, and the improvements observed are due to the learning technique.

5) Window size: Our preliminary results on window sizes for proposal generation show that combining the results from multiple scales gives far superior performance compared to single scale. The results at different scales are shown in Figure 9.

D. Required annotations per video

The videos in THUMOS'14 dataset have 15 action instances per video on average. The actions are not distributed evenly among videos and the dataset has a standard deviation of 24 with respect to number of action instances. The action instances per video range from 1 to 128 (Figure 10). The total labeled action instances in the training set if we fix the maximum action instances per video are shown in Figure 11. We check whether annotations for all instances are needed to get an effective action proposal network. We compare the F1-score for maximum annotations per video ranging from 1 to 128. After a maximum limit of 6 annotations per video, the F1-score has low variance (Figure 11). The network is able to identify the unlabeled key frames effectively. We believe that not all annotations are necessary to achieve good performance.

We train our network, PUNet, with maximum of 6 annotations per video and report results in Table VI. To validate the generalizability of limiting annotations, we train the state of the art weakly supervised action localization networks with limited annotations. BaSNet [27] and BUME [28] are trained with the reduced video size and the results show a 0.9% and 2.5% reduction in mAP, while only utilizing one third action instances. The results are shown in Table VII.

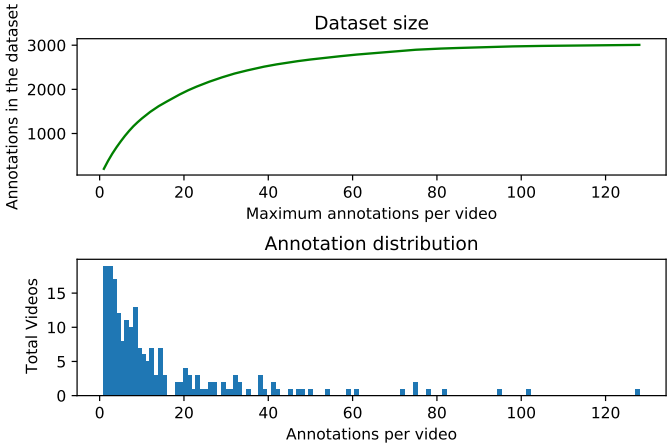


Fig. 10. THUMOS'14 dataset distribution. The number of annotations per video vary drastically ranging from one action instance to 128. Limiting the maximum number of annotations per video reduces the annotations required significantly.

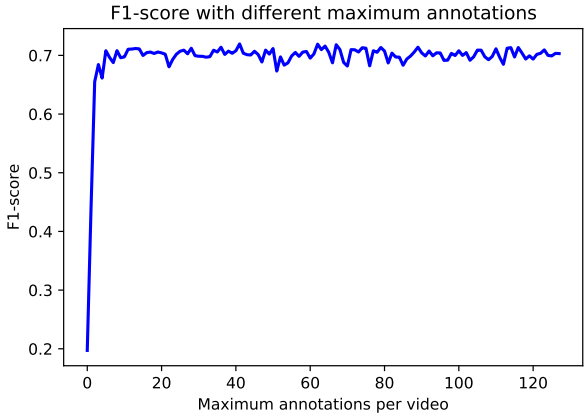


Fig. 11. Effect of changing the number of annotations per video on the classifier performance. After 6 annotations per video, the performance does not change much and the standard deviation reduces. The mean value of F1-score from 1-128 annotations is 0.69 $\pm$ + 0.05, and mean F1-score from 6-128 annotaions is 0.70 $\pm$ 0.008. Our method does not need all the annotations to perform well.

TABLE VI
Effect of using limited annotations per segment and limiting the segment length on proposal generation. We set the maximum annotations per video to 6. The average recall is reported for 50 and 100 proposals. The action instances are reduced by one-third while giving comparable average recall.

Input	Action instances	AR@50	AR@100
Partial	946	31.27	36.84
Whole	3007	33.34	41.10

TABLE VII
Effect of using limited annotations per segment and limiting the segment length on action localization for THUMOS'14 dataset. We set the maximum annotations per video to 6. The action instances needed reduce by one-third while the performance only decreases by 0.9% and 2.5% for BaSNet and BUME.

Input	Action instances	Method	mAP
Partial	946	BaSNet	26.1
		BUME	28.2
Whole	3007	BaSNet	27.0
		BUME	30.7

TABLE VIII
Generalization evaluation of PUNet on THUMOS'14 dataset. Action classes are removed from the training set and the resulting model is evaluated on the full test set (seen + unseen classes) containing 20 classes.

# classes in training set	AR@50	AR@100
17	32.7	38.3
18	33.1	39.7
19	33.2	40.8
20	33.3	41.1

E. Generalizability of proposals

We evaluate the generalization ability of PUNet by testing its performance on unseen action classes. We randomly leave one to three classes from our training set and test on our test set containing all 20 classes of THUMOS'14 data. As shown in table VIII, there is only a slight performance decrease when testing on unseen classes and the method is able to generate high quality proposals on unseen classes.

F. Qualitative analysis

The qualitative analysis of our approach for key frame annotation is shown in Figure 1. We chose two actions $BaseballPitch$ and $CleanandJerk$ from THUMOS'14 to evaluate our method. The window size is kept to 32 frames. The GT denotes groundtruth segments and the labels denote the key frame inputs to our network. Without any postprocessing, our proposal evaluation model is able to capture the full extent of the temporal duration and not just the key frames. The proposals generated for action $CricketBowling$ and $CleanandJerk$ are shown in Figure **??**.

V. Conclusion

In this work, we investigate using key frame level supervision for training temporal action proposal model. We propose a method which requires labeling less annotations per video and PU-learning. We test our approach on two untrimmed datasets. Compared to fully supervised methods, our approach is able generate proposals with high recall and high temporal overlap. Experimental evaluation on THUMOS'14 and ActivityNet v1.2 shows that: (i) Using a key frame annotation gives comparable performance to using fully supervised annotation which use start and end annotations, (ii) All action instances from one video are not necessary to achieve good detection results, (iii) Our results are comparable to the state of the art methods. We conclude that annotation effort can be significantly reduced by labeling key frames and for long untrimmed videos, only limited number of action instances need to be labeled and trained to achieve similar results.

References

[1] 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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 1130–1139.

[2] Y. Liu, L. Ma, Y. Zhang, W. Liu, and S.-F. Chang, "Multi-granularity generator for temporal action proposal," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 3604–3613.

[3] F. C. Heilbron, W. Barrios, V. Escorcia, and B. Ghanem, "Scc: Semantic context cascade for efficient action detection," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2017, pp. 3175–3184.

[4] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein *et al.*, "Imagenet large scale visual recognition challenge," *International journal of computer vision*, vol. 115, no. 3, pp. 211–252, 2015.

[5] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in neural information processing systems*, 2015, pp. 91–99.

[6] P. Weinzaepfel, Z. Harchaoui, and C. Schmid, "Learning to track for spatio-temporal action localization," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 3164–3172.

[7] G. Gkioxari and J. Malik, "Finding action tubes," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 759–768.

[8] S. Yeung, O. Russakovsky, N. Jin, M. Andriluka, G. Mori, and L. Fei-Fei, "Every moment counts: Dense detailed labeling of actions in complex videos," *International Journal of Computer Vision*, 2017.

[9] D. Moltisanti, M. Wray, W. Mayol-Cuevas, and D. Damen, "Trespassing the boundaries: Labeling temporal bounds for object interactions in egocentric video," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2886–2894.

[10] S. Satkin and M. Hebert, "Modeling the temporal extent of actions," in *European conference on computer vision*. Springer, 2010, pp. 536–548.

[11] O. Duchenne, I. Laptev, J. Sivic, F. Bach, and J. Ponce, "Automatic annotation of human actions in video," in *2009 IEEE 12th International Conference on Computer Vision*. IEEE, 2009, pp. 1491–1498.

[12] K. Schindler and L. Van Gool, "Action snippets: How many frames does human action recognition require?" in *2008 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2008, pp. 1–8.

[13] X. Yang and Y. Tian, "Effective 3d action recognition using eigenjoints," *Journal of Visual Communication and Image Representation*, vol. 25, no. 1, pp. 2–11, 2014.

[14] D. Moltisanti, S. Fidler, and D. Damen, "Action recognition from single timestamp supervision in untrimmed videos," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[15] P. Mettes, J. C. Van Gemert, and C. G. Snoek, "Spot on: Action localization from pointly-supervised proposals," in *European conference on computer vision*. Springer, 2016, pp. 437–453.

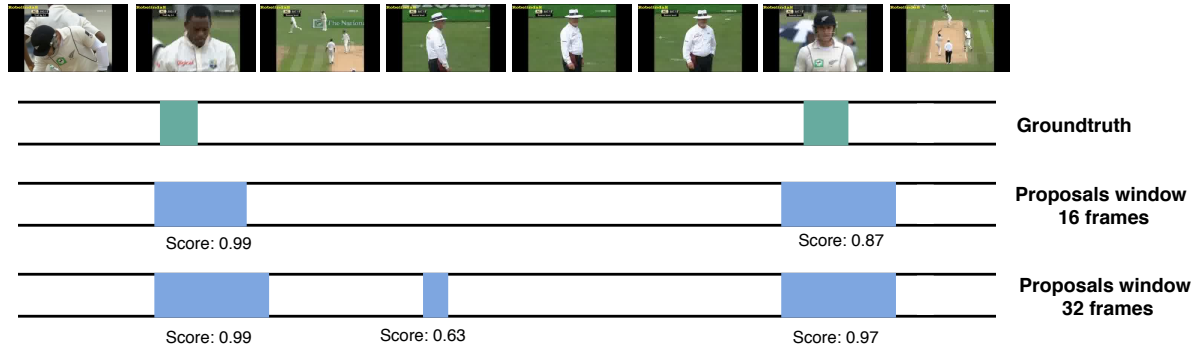
[16] J. Bekker and J. Davis, "Learning from positive and unlabeled data: a survey." *Mach. Learn.*, vol. 109, no. 4, pp. 719–760, 2020.

[17] T. Lin, X. Zhao, H. Su, C. Wang, and M. Yang, "Bsn: Boundary sensitive network for temporal action proposal generation," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 3–19.

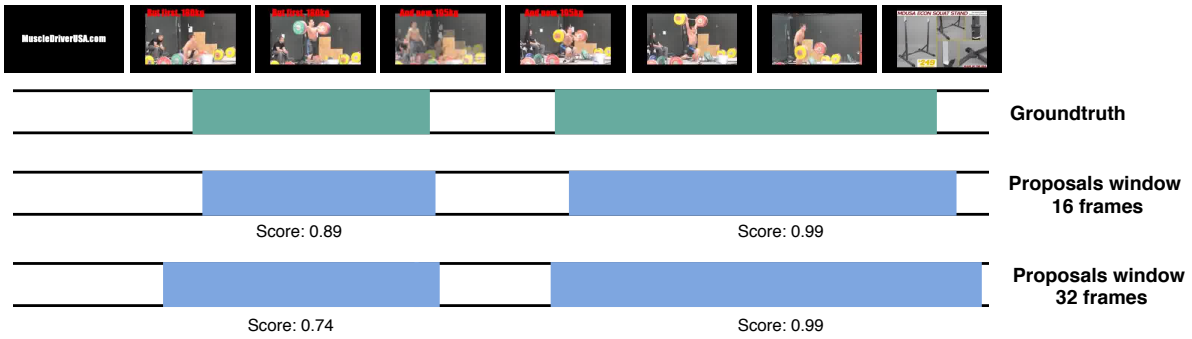
[18] Y. Zhao, Y. Xiong, L. Wang, Z. Wu, X. Tang, and D. Lin, "Temporal action detection with structured segment networks," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2914–2923.

[19] F. Long, T. Yao, Z. Qiu, X. Tian, J. Luo, and T. Mei, "Gaussian temporal awareness networks for action localization," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 344–353.

[20] R. Zeng, W. Huang, M. Tan, Y. Rong, P. Zhao, J. Huang, and C. Gan, "Graph convolutional networks for temporal action localization," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.



(a) Cricket bowling



(b) Clean and Jerk

Fig. 12. The proposals generated using PUNet at window sizes 16 and 32 frames. PUNet is able to generate proposals encompassing the boundaries of the action instance with high recall. The false detections have a lower score compared to actual action instances.

[21] L. Wang, Y. Qiao, and X. Tang, "Action recognition and detection by combining motion and appearance features," *THUMOS14 Action Recognition Challenge*, vol. 1, no. 2, p. 2, 2014.

[22] A. Gaidon, Z. Harchaoui, and C. Schmid, "Temporal localization of actions with actoms," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 11, pp. 2782–2795, 2013.

[23] S. Buch, V. Escorcia, C. Shen, B. Ghanem, and J. Carlos Niebles, "Sst: Single-stream temporal action proposals," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2017, pp. 2911–2920.

[24] T. Lin, X. Liu, X. Li, E. Ding, and S. Wen, "Bmn: Boundary-matching network for temporal action proposal generation," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 3889–3898.

[25] L. Wang, Y. Xiong, D. Lin, and L. Van Gool, "Untrimmednets for weakly supervised action recognition and detection," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2017, pp. 4325–4334.

[26] K. K. Singh and Y. J. Lee, "Hide-and-seek: Forcing a network to be meticulous for weakly-supervised object and action localization," in *2017 IEEE international conference on computer vision (ICCV)*. IEEE, 2017, pp. 3544–3553.

[27] P. Lee, Y. Uh, and H. Byun, "Background suppression network for weakly-supervised temporal action localization." in *AAAI*, 2020, pp. 11 320–11 327.

[28] P. Lee, J. Wang, Y. Lu, and H. Byun, "Background modeling via uncertainty estimation for weakly-supervised action localization," *arXiv preprint arXiv:2006.07006*, 2020.

[29] B. Liu, W. S. Lee, P. S. Yu, and X. Li, "Partially supervised classification of text documents," in *ICML*, vol. 2, 2002, pp. 387–394.

[30] B. Liu, Y. Dai, X. Li, W. S. Lee, and P. S. Yu, "Building text classifiers using positive and unlabeled examples," in *Third IEEE International Conference on Data Mining*. IEEE, 2003, pp. 179–186.

[31] D. H. Fusilier, M. Montes-y Gómez, P. Rosso, and R. G. Cabrera, "Detecting positive and negative deceptive opinions using pu-learning," *Information processing & management*, vol. 51, no. 4, pp. 433–443, 2015.

[32] Y.-G. Jiang, J. Liu, A. R. Zamir, G. Toderici, I. Laptev, M. Shah, and R. Sukthankar, "Thumos challenge: Action recognition with a large number of classes," 2014.

[33] F. Caba Heilbron, V. Escorcia, B. Ghanem, and J. Carlos Niebles, "Activitynet: A large-scale video benchmark for human activity understanding," in *Proceedings of the ieee conference on computer vision and pattern recognition*, 2015, pp. 961–970.

[34] V. Escorcia, F. C. Heilbron, J. C. Niebles, and B. Ghanem, "Daps: Deep action proposals for action understanding," in *European Conference on Computer Vision*. Springer, 2016, pp. 768–784.

[35] F. Caba Heilbron, J. Carlos Niebles, and B. Ghanem, "Fast temporal activity proposals for efficient detection of human actions in untrimmed videos," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 1914–1923.

[36] Z. Shou, D. Wang, and S.-F. Chang, "Temporal action localization in untrimmed videos via multi-stage cnns," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 1049–1058.

[37] J. Gao, Z. Yang, K. Chen, C. Sun, and R. Nevatia, "Turn tap: Temporal unit regression network for temporal action proposals," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 3628–3636.

[38] Z. Shou, J. Chan, A. Zareian, K. Miyazawa, and S.-F. Chang, "Cdc: Convolutional-de-convolutional networks for precise temporal action localization in untrimmed videos," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 5734–5743.

[39] P. Nguyen, T. Liu, G. Prasad, and B. Han, "Weakly supervised action localization by sparse temporal pooling network," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 6752–6761.

[40] Z. Shou, H. Gao, L. Zhang, K. Miyazawa, and S.-F. Chang, "Autoloc: Weakly-supervised temporal action localization in untrimmed videos," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 154–171.

[41] D. Liu, T. Jiang, and Y. Wang, "Completeness modeling and context separation for weakly supervised temporal action localization," in

Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 1298–1307.

[42] L. Huang, Y. Huang, W. Ouyang, L. Wang *et al.*, "Relational prototypical network for weakly supervised temporal action localization," 2020.

[43] S. Paul, S. Roy, and A. K. Roy-Chowdhury, "W-talc: Weakly-supervised temporal activity localization and classification," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 563–579.

[44] J. Carreira and A. Zisserman, "Quo vadis, action recognition? a new model and the kinetics dataset," in *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 6299–6308.

2

Datasets

Action recognition methods rely on trimmed videos where the video only consists of the relevant action frames. Most real world videos are unconstrained and consist of multiple action instances, referred to as untrimmed videos. Untrimmed videos are longer, unconstrained and contain noise and background frames. This poses a need to localize the frames which contain the action and classify them. Several datasets have been collected containing untrimmed videos to help in action localization research. To ensure that the model generalizes well and is able to learn from different types of data, it is important to test on more than one dataset. In this work, two untrimmed datasets have been used: THUMOS'14 [4] and ActivityNet 1.2 [1].

2.1. Dataset selection

THUMOS'14 dataset consists of 200 validation and 210 test videos. Each video consists of multiple action instances with ∼15 action instances per video on average with a standard deviation of 24. The videos can contain action from multiple classes.

ActivityNet 1.2 dataset consists of 100 action classes with 4819 training and 2383 validation videos, containing ∼1.5 action instances per video. The videos are shorter in length compared to THUMOS'14 dataset while containing bigger range of action classes. Using these two datasets for evaluation allows us to determine:

- THUMOS'14: The ability of the model to detect actions in long videos consisting of multiple action instances.
- ActivityNet 1.2: The ability of model to detect actions belonging to many classes

The characteristics of both datasets have been summarized in Table 2.1.

Table 2.1: Statistics of THUMOS'14 and ActivityNet 1.2 dataset. Both datasets contain multiple actions per video. ActivityNet has more number of classes while THUMOS'14 dataset contains more action instances per video.

Dataset	Set	No. of classes	No. of videos	No. of actions	Average actions per video
THUMOS'14	*Train*	20	200	3007	15.03
	Test	20	210	3307	15.74
ActivityNet 1.2	*Train*	100	4819	7151	1.48
	Test	100	2383	3583	1.50

3

Data preprocessing

In key frame based annotation, one frame is labeled for each action instance. Existing data annotations are converted into key frame level annotations. This section discusses the annotation process and the preprocessing of input data.

3.1. Annotation

For a given video, if an action is from t_1 to t_2, a single frame is labeled during this instance duration which is referred to as key frame. Key frame position is tested on THUMOS'14 by labeling start, end, random and midpoint of the duration as key frame. Midpoint and random point give similar performance for foreground and background classification (Table 3.1). Midpoint is picked as key frame for further experiments.

Multiple window scales are used for experimentation. For a given window scale, the video is divided into non overlapping chunks. If the key frame is present in the frames located within the window, the window is assigned a positive label. The rest of the windows are unlabeled. The dataset statistics for each window scale are presented in Table 3.2. For the test set, all frames in the action duration are assigned as positive.

3.2. Feature extraction

I3D [2] pretrained on Kinetics dataset is used as a feature extractor for the input videos. The videos are formatted to 25 FPS. The frames and optical flows are extracted at 25 FPS. The rgb and optical flow inputs are given to the I3D network separately. Features are extracted from Mixed_5c layers. Feature is extracted for every chunk of 16 frames. The dimension of feature vector is 1024. The RGB and flow features are concatenated to get a 2048 dimension feature vector.

3.2.1. Two-Stream Inflated 3D ConvNet (I3D)

I3D [2] combines the benefits of 3D convnets [7] and two stream networks [6]. Existing 2D image classification models are converted into 3D convnet by inflating all the filters and pooling kernels in a 2D architecture with an additional temporal dimension (Figure 3.1). This allows converting 2D classification models into 3D by training multiple frames. The pretrained weights from 2D networks can also be used

Table 3.1: Results of using different frames as the key frame. Middle frame and random frame give the best results.

Key frame type	F1-score
Midpoint	0.64
Random point	0.62
3 Random points	0.61
Start point	0.54
End point	0.44

Table 3.2: Number of positive and unlabeled samples with different window scales for THUMOS'14 dataset.

Window scale (frames)	Seconds	Positive samples	Unlabeled samples	Ratio
16	1	2848	62143	1:22
32	2	2779	29668	1: 11
48	3	2717	18873	1:7
64	4	2657	13519	1:5
80	5	2583	10337	1:4

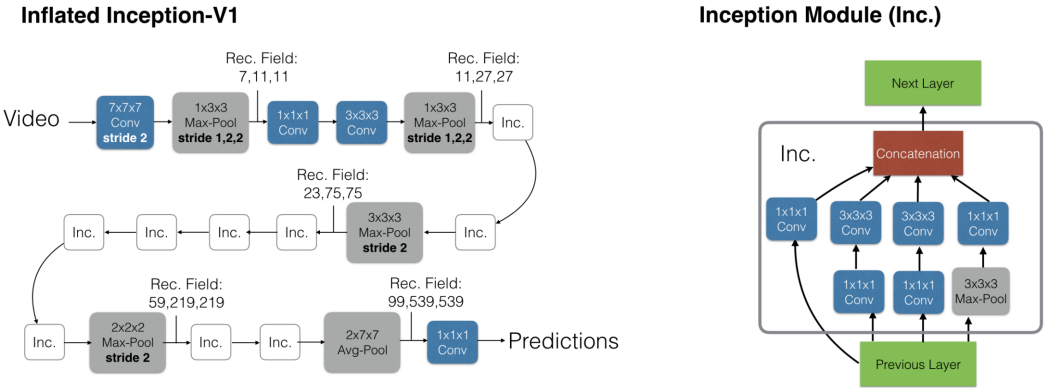


Figure 3.1: The Inflated Inception-V1 architecture (left) and its detailed inception submodule (right) [2]

by repeating the weights of 2D filters N times in the time dimension. The inflated filters are N x N x N dimension instead of N x N.

4

PU learning

In an ideal learning setting, the training data is fully labeled. However, obtaining labeled data is an expensive and time consuming task. Positive Unlabeled (PU) learning allows the model to learn from data by explicitly using the unlabeled data in its training process. In PU learning setting, the model has access to few positively labeled examples and the rest of the data is unlabeled. The unlabeled data can consist of both positives and negatives.

PU learning approach used in this thesis is based on [3]. The training data consists of video frames and each frame is either positive or unlabeled. Instead of labeling the full duration of an action, a single frame is labeled inside the window of the action instance. This approach has the following advantages:

1. Reduces the labeling effort
2. Reduces the subjective bias of labeling
3. The start and end points are more prone to error while labeling. By using a point with-in the action window, the probability of this error is reduced.

4.1. PU learning algorithm

The modified PU learning algorithm proposed by Fusilier et al. [3] is used. A classifier in trained by considering all unlabeled data as negatives. The trained classifier is then used to predict the unlabeled data. The predicted negatives with a confidence higher than θ are chosen as reliable negatives (RN). RN set is refined iteratively by training P vs RN until no samples in the RN set are classified as positives. The steps are listed in Algorithm 1.

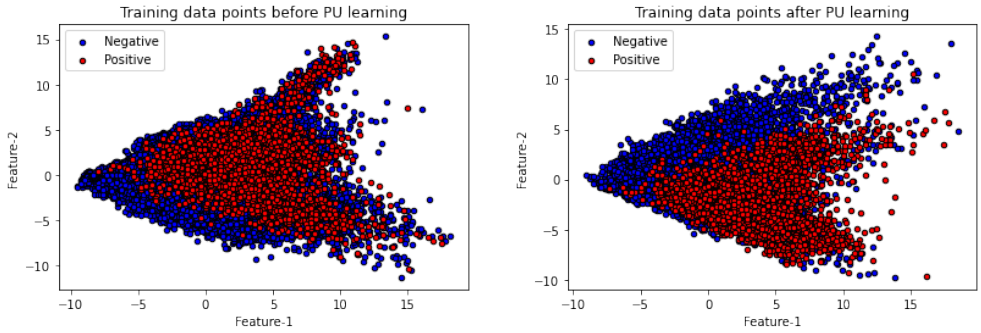
Algorithm 1: PU learning algorithm

Result: Binary Classifier C_i
$i \leftarrow 1$;
$C_i \leftarrow Classifier(P, U)$
$U_i^L \leftarrow C_i(U)$
$Q_i \leftarrow threshold\_negatives(U_i^L, \theta)$
$RN_i \leftarrow Q_i$
$Q_0 \leftarrow Q_i$
while $| Q_i | \leq | Q_{i-1} | and | P | < | RN_i |$ **do**
  $i \leftarrow i + 1$
  $C_i \leftarrow Classifier(P, RN_{i-1})$
  $RN_i^L \leftarrow C_i(RN_{i-1})$
  $Q_i \leftarrow negatives(RN_i^L)$
  $RN_i \leftarrow Q_i$
end

4.2. Qualitative results of PU learning

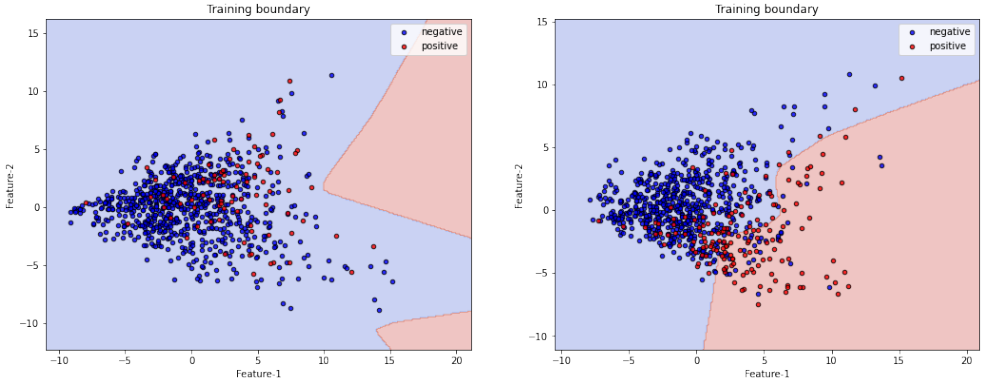
Pu learning method helps in fitting the decision boundary in imbalanced class data. For visualization purposes, the feature dimension is reduced to two using PCA. Randomly chosen samples from training and test set are plotted to show the effect on decision boundary after training a classifier using PU learning strategy. The training process uses all the data points.

Figure 4.1a shows the positive and unlabeled data samples for THUMOS'14 dataset. After refining the unlabeled set using PU learning, positive and reliable negatives are shown in Figure 4.1b. The effect on decision boundary after training classifier on data obtained before and after PU learning is shown in Figure 4.2 and 4.3 for training and testing data respectively.



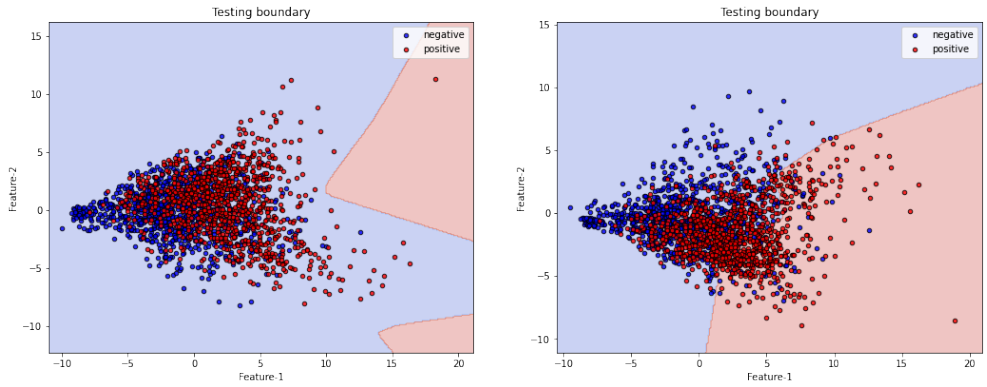
(a) Data points without PU learning (b) Data points after applying PU learning

Figure 4.1: Before PU learning (Figure a), the dataset is imbalanced and consists of less positive samples compared to unlabeled samples. After applying PU learning algorithm, the unlabeled set reduces in size and is replaced by reliable negatives (Figure b).



(a) No PU - training (b) PU - training

Figure 4.2: 200 samples are shown from the training set. (a) The decision boundary without PU learning does not fit the data points. (b) After PU learning, the decision boundary does a better job at separating negatives and positives,



(a) No PU - test (b) PU test

Figure 4.3: 500 samples are shown from the test set. (a) The decision boundary without PU learning does not fit the data points. (b) After PU learning, the decision boundary does a better job at separating negatives and positives,

4.3. Quantitative results of PU learning

The f1-score of the PU learning network shown in the previous section is reported for the test set in Table 4.1. Due to low dimension and class imbalance, these numbers are lower for dimension 2 than for the full dimension. However, the performance improvement due to the proposed method is noticeable. The results for dimension 2, correspond to the decision boundary shown in figure 4.3b.

Table 4.1: The performance of network before and after refining the dataset using PU learning. The results are reported for feature dimensions 2 and 2048. PU learning significantly improves the performance.

Dimension	Set up	F1 score
2	*Baseline (No PU)*	0.0004
	PU learning	0.61
2048	*Baseline (No PU)*	0.14
	PU learning	0.68

5

Alternative attemps

5.1. Siamese network

One of the post processing method proposed in this work uses a Siamese Network. A pairwise learning approach is used to compute similarity between given pairs of input frames. The input pairs are selected as a triplet comprising of an anchor frame and postive and negative samples for that anchor frame. The dissimilarity should be high between the anchor frame and negative sample. Similarly, the dissimilarity should be low between anchor frame and positive sample. This phenomenon is implemented through the triplet loss:

$$L = max(d(a,p) - d(a,n) + margin, 0)$$

where d is the distance function, a is the anchor point, n is negative point and p is positive point. The loss pushes $d(a,p)$ to 0 and $d(a,n)$ to be greater than $d(a,p) + margin$.

5.1.1. Boundary refinement

After training the network with key frames labels, during inference we obtain action frames on the testing video. The network is able to detect all action frames, not just the key frames. To localize the temporal segment of an action, we use matching algorithms to refine the boundaries of the segment. Siamese network is used with triplet loss. For a given anchor, frames that are within a small window of the anchor frame are assigned as positive input. The negative inputs are selected by either randomly selecting a frame from outside the window or choosing a random frame from a different video. Once Siamese network is trained, similarity is calculated between an action frame and frames on its either sides, if the similarity is greater than a threshold, those frames are added to the action segment. The method with siamese network based boundary refinement is shown is Figure 5.1.

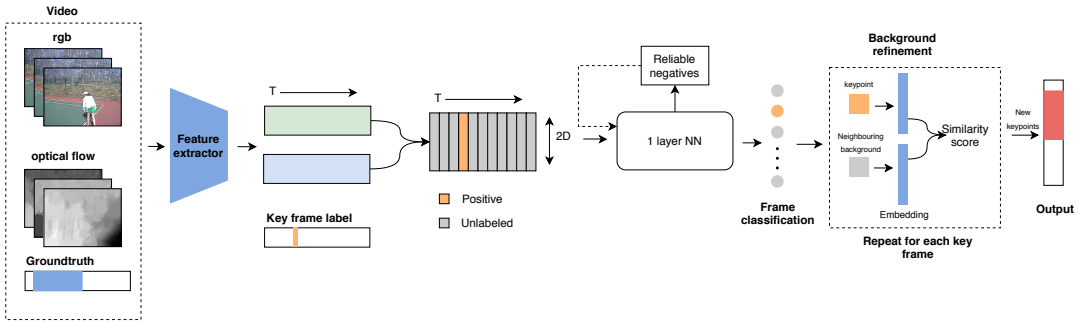


Figure 5.1: Overview of our proposed approach with Siamese matching. One frame is labeled for each action instance. The model is trained using PU learning technique. For each detected action frame, the boundaries are compared with adjacent background frames using a Siamese network. Similar frames are merged with the action segment to get the final result.

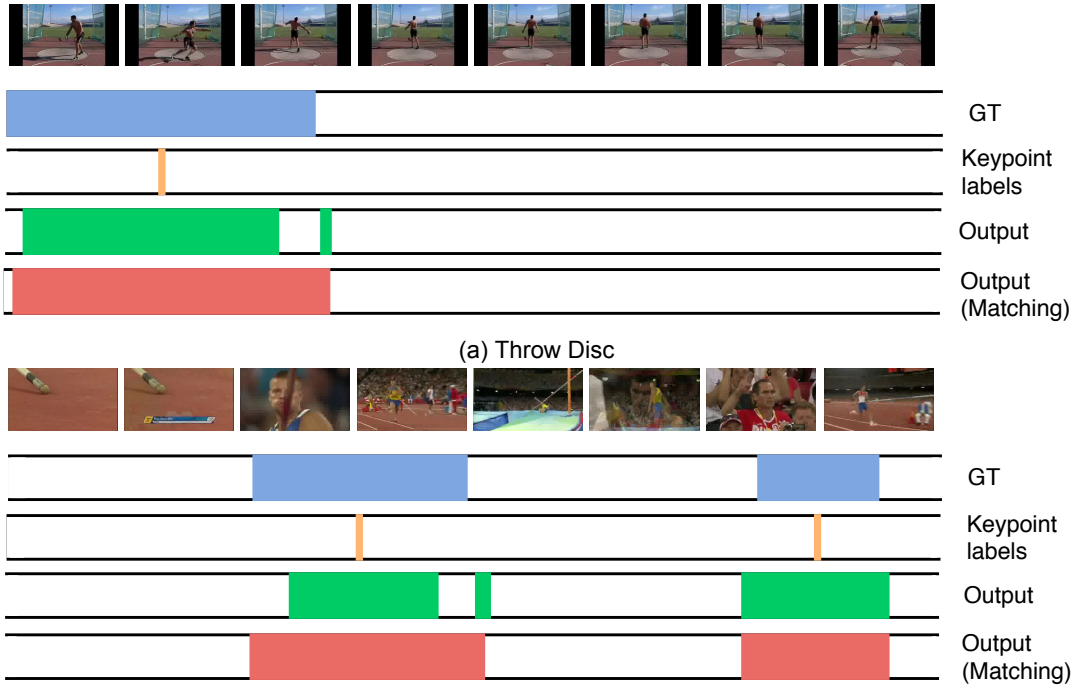


Figure 5.2: Detected segments on THUMOS'14 dataset with Background refinement. GT indicates the groundtruth, keypoint labels are our network input. The last row represents the results after background refinement. The refinement step helps in combining segments from same action instance and improves the results for longer segments.

5.1.2. Qualitative results

The additional matching is able to refine the detected temporal durations. For longer durations, network has a lower overlap with groundtruth. By using a siamese network to find similar frames, the localization performance of the network improves (Figure 5.2). The results are shown on $ThrowDisc$ and $PoleVault$ actions of THUMOS'14 dataset.

5.1.3. Discussion

The refinement approach while giving good results, requires some heuristics and knowledge about the dataset to select the pairs in Siamese network. It is also more computationally expensive. The performance after using multiple window scales and siamese network for matching is similar. But due to the less computation required for the multiple window scales set up, the siamese matching was removed as a post processing step.

6

Discussion

In this work, an alternative approach for labeling the temporal annotations of the action in videos has been proposed. The temporal boundaries obtained using start and end points are subjective and prone to error [5]. The labeling process is also taxing and requires watching the video multiple times to detect precise boundaries [8]. Whereas labeling a single frame within the action instance is less time consuming and taxing.

6.1. Recommendations

6.1.1. Datasets

The results have been reported on two popular video action datasets. The thesis work does not test the performance of the network on egocentric datasets. The results are reported for ActivityNet 1.2 and not the latest version ActivityNet 1.3[1]. This decision was made as ActivityNet 1.3 has twice the amount of data as its predecessor version, while both have the same complexity of videos (length, average action instances). To optimize the time and resources available, a decision was made to use ActivityNet 1.2. For future work, the work can be tested on egocentric datasets and ActivityNet 1.3.

6.1.2. Multiclass set up

The PU learning problem is treated as a binary class problem. The proposals are classified using a preexisting classier. The preliminary experiments using multi class PU learning gave good accuracy results 73% overall but the detection results were poor. On further inspection, it was noticed that classes belonging to shorter videos with less key frame annotations available are the ones that perform poorly. This is an area that can be experimented with in more detail during future work by limiting the maximum annotations for each class to avoid this class imbalance.

[1] http://activity-net.org/index.html

Bibliography

[1] Fabian Caba Heilbron, Victor Escorcia, Bernard Ghanem, and Juan Carlos Niebles. Activitynet: A large-scale video benchmark for human activity understanding. In *Proceedings of the ieee conference on computer vision and pattern recognition*, pages 961–970, 2015.

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

[3] Donato Hernández Fusilier, Manuel Montes-y Gómez, Paolo Rosso, and Rafael Guzmán Cabrera. Detecting positive and negative deceptive opinions using pu-learning. *Information processing & management*, 51(4):433–443, 2015.

[4] Yu-Gang Jiang, Jingen Liu, A Roshan Zamir, George Toderici, Ivan Laptev, Mubarak Shah, and Rahul Sukthankar. Thumos challenge: Action recognition with a large number of classes, 2014.

[5] Davide Moltisanti, Michael Wray, Walterio Mayol-Cuevas, and Dima Damen. Trespassing the boundaries: Labeling temporal bounds for object interactions in egocentric video. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2886–2894, 2017.

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

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

[8] Hang Zhao, Antonio Torralba, Lorenzo Torresani, and Zhicheng Yan. Hacs: Human action clips and segments dataset for recognition and temporal localization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 8668–8678, 2019.