Supplementary Material

Linjiang Huang[1,2] Liang Wang[3] Hongsheng Li[1,2] *
[1]CUHK-SenseTime Joint Laboratory, The Chinese University of Hong Kong
[2]Centre for Perceptual and Interactive Intelligence, Hong Kong
[3]Institute of Automation, Chinese Academy of Sciences
ljhuang524@gmail.com, wangliang@nlpr.ia.ac.cn, hsli@ee.cuhk.edu.hk

In this supplementary material, more details about our methods are presented first. Then we report additional experimental results to further validate our network design. At last, we show more qualitative detection results, some failure cases and feature visualizations.

1. Architecture of the Classification Head

The classification head is mainly used to generate TCAMs, it can be any existing WSTAL methods. To generate high quality TCAMs and improve the lower bound of our method, we use the recent method FAC-Net [6] as the classification head for its simple pipeline and strong performance.

We simply revisit the FAC-Net in this section. As the most weakly supervised methods, FAC-Net is based on the fixed-weight I3D backbone network, on which a small learnable network (*e.g.*, 1×1 convolutional network) is appended to learn the video snippet features $\mathbf{F} \in \mathbb{R}^{l \times d}$. Besides, FAC-Net contains a foreground classifier $\mathbf{W}_f$ and an action classifier $\mathbf{W}_a$. Given the video snippet features $\mathbf{F}$, there are three classification heads are appended on. The first classification head is a class-agnostic attention (CA) head. It first calculates the cosine similarity between $\mathbf{W}_f$ and $\mathbf{F}$ to get the foreground attention weights $\boldsymbol{\lambda}_f \in \mathbb{R}^l$. The foreground attention weights are used to aggregate snippet features $\mathbf{F}$ into a video level feature $\bar{\mathbf{F}}$, which is used to calculate the video-level prediction with $\mathbf{W}_a$. The second classification head is a multiple instance learning (MIL) head. It calculates the cosine similarity between $\mathbf{W}_a$ and $\mathbf{F}$ to obtain the snippet-wise class logits $\mathbf{S} \in \mathbb{R}^{l \times (c+1)}$. The TCAMs $\mathbf{T}$ and the class-wise attention scores $\boldsymbol{\lambda}_w$ are obtained by applying softmax to $\mathbf{S}$ along category dimension and temporal dimension, respectively. The video-level prediction of the MIL head is obtained by aggregating $\mathbf{S}$ with $\boldsymbol{\lambda}_w$. There is another classification head, namely class-wise foreground classification head. However, as shown in Figure 1, we do not use this head in our method, since the former two heads already enable our method to achieve a high baseline performance. Therefore, in our method, for each classification head, there are only two video-level classification losses, which corresponds to the CA branch and the MIL branch, respectively. We weightedly combine the two video-level classification losses as $\mathcal{L}_{cls} = \mathcal{L}_{ca} + \gamma \mathcal{L}_{mil}$, where γ denotes the balancing hyper-parameter, which is set as 0.2 in our method.

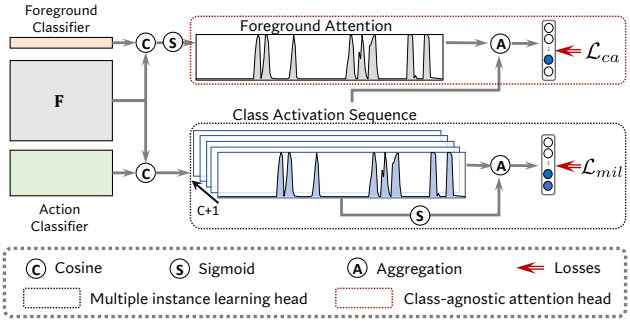

Figure 1. The overview of the classification head used in our method, including a class-agnostic attention head and a multiple instance learning head in FAC-Net [6].

There are some other modifications upon the FAC-Net. First, we use the sigmoid rather than the softmax function to obtain normalized foreground scores. This setting enables our method to use the attention normalization term [14] to obtain highly confident representative snippets. Second, we do not use the hybrid attention strategy, which is designed to alleviate the discrepancy between classification and detection.

2. Bipartite Random Walk Module

First, we briefly introduce the random walk operation. Random walks are one of the most widely known and used methods in graph theory [9], which led to the development of PageRank [10], personalized PageRank [1], DeepWalk [12], DeeperGCN [8], *etc*. Let G denotes an undirected graph. There is a transition matrix $\mathbf{A}$ of the graph G. Given the feature $\mathbf{F}$ of graph nodes, the random walk operation can be done via a simple matrix multiplication operation

*Corresponding author.

$\mathbf{F}^{(1)} = \mathbf{AF}$.

In [2, 13], they use the random walk operation to propagate information between image pixels or probe images for semantic segmentation and person re-identification. They propose to weighted combine $\mathbf{AF}$ with the input $\mathbf{F}$ to make $\mathbf{F}^{(1)}$ be not deviate too far away from $\mathbf{F}$, which can be formulated as

$$\mathbf{F}^{(1)} = w\mathbf{AF} + (1-w)\mathbf{F}, \tag{1}$$

where w is a balancing hyper-parameter. This process can be conducted multiple times until convergence

$$\mathbf{F}^{(t)} = w\mathbf{AF}^{(t-1)} + (1-w)\mathbf{F}. \tag{2}$$

As $t \to \infty$, there is an approximate inference as

$$\mathbf{F}^{(\infty)} = (1-w)(\mathbf{I} - w\mathbf{A})^{-1}\mathbf{F}, \tag{3}$$

where $\mathbf{I}$ denotes an identity matrix.

As state in our main manuscript, at the t-th iteration, the bipartite random walk operation can be formulated as (for simplicity, we omit the upper-script *, which is a or e corresponding to the online representative snippets or the offline representative snippets)

$$\boldsymbol{\mu}^{(t)} = w\,\mathrm{Norm}_1(\mathbf{Z})^\top \mathbf{F}^{(t-1)} + (1-w)\boldsymbol{\mu}^{(0)}, \tag{4}$$

$$\mathbf{F}^{(t)} = w\mathbf{Z}\boldsymbol{\mu}^{(t)} + (1-w)\mathbf{F}^{(0)}. \tag{5}$$

Substitute Equation (4) into Equation (5), we can obtain

$$\begin{aligned}\mathbf{F}^{(t)} &= w^2\mathbf{Z}\,\mathrm{Norm}_1(\mathbf{Z})^\top\mathbf{F}^{(t-1)} + (1-w)w\mathbf{Z}\boldsymbol{\mu}^{(0)} \\ &\quad +(1-w)\mathbf{F}^{(0)} \\ &= w^2\mathbf{Z}\,\mathrm{Norm}_1(\mathbf{Z})^\top\mathbf{F}^{(t-1)} \\ &\quad +(1-w)(w\mathbf{Z}\boldsymbol{\mu}^{(0)} + \mathbf{F}^{(0)}).\end{aligned} \tag{6}$$

In the following equations, we denote $\mathbf{Z}\,\mathrm{Norm}_1(\mathbf{Z})^\top$ as $\mathbf{R}$. Therefore, expanding Equation (6) leads to

$$\mathbf{F}^{(t)} = (w^2\mathbf{R})^t\mathbf{F}^{(0)} + (1-w)\sum_{i=0}^{t-1}(w^2\mathbf{R})^i(w\mathbf{Z}\boldsymbol{\mu}^{(0)} + \mathbf{F}^{(0)}). \tag{7}$$

As $t \to \infty$, since elements in $\mathbf{R}$ and $w \in [0, 1]$,

$$\lim_{t\to\infty}(w^2\mathbf{R})^t\mathbf{F}^{(0)} = \mathbf{0}. \tag{8}$$

For $\sum_{i=0}^{t-1}(w^2\mathbf{R})^i$, the matrix series can be expanded as

$$\lim_{t\to\infty}\sum_{i=0}^{t-1}(w^2\mathbf{R})^i = (\mathbf{I} - w^2\mathbf{R})^{-1}. \tag{9}$$

Therefore, Equation (6) can be formulated as

$$\mathbf{F}^{(\infty)} = (1-w)(\mathbf{I} - w^2\mathbf{Z}\,\mathrm{Norm}_1(\mathbf{Z})^\top)^{-1}(w\mathbf{Z}\boldsymbol{\mu}^{(0)} + \mathbf{F}^{(0)}). \tag{10}$$

Table 1. Evaluation of the attention normalization loss.

Method	$\mathcal{L}_{att}$	AVG
Baseline	✗	35.6
	✓	36.8
+ representative snippets	✗	40.2
	✓	40.1
+ pseudo label supervision	✗	42.0
	✓	44.2
+ memory bank	✗	42.5
	✓	45.1

When $\boldsymbol{\mu}^{(0)} = \mathbf{F}^{(0)}$, that is, the propagation is directly conducted between the features of $\mathbf{F}$, there exists $\mathrm{Norm}_1(\mathbf{Z})^\top = \mathbf{Z}$, because the bi-directional affinities between $\mathbf{F}^{(0)}$ and $\mathbf{F}^{(0)}$ should be the same. At this time, Equation (10) can be rewrote as

$$\begin{aligned}\mathbf{F}^{(\infty)} &= (1-w)(\mathbf{I} - w^2\mathbf{Z}^2)^{-1}(w\mathbf{Z}\mathbf{F}^{(0)} + \mathbf{F}^{(0)}), \\ &= (1-w)(\mathbf{I} - w\mathbf{Z})^{-1}(\mathbf{I} + w\mathbf{Z})^{-1}(w\mathbf{Z}\mathbf{F}^{(0)} + \mathbf{F}^{(0)}), \\ &= (1-w)(\mathbf{I} - w\mathbf{Z})^{-1}\mathbf{F}^{(0)},\end{aligned} \tag{11}$$

which is the same as Equation (3).

3. More Implementation Details

For each snippet, a 2048-d feature is extracted by the I3D pre-trained on Kinetics-400 [3]. The following one 1×1 convolutional layer outputs 2048-d features. Besides, the number of online representative snippets for each video is 8, while the number of representative snippets for each class in the memory bank is 5.

4. Additional Ablation Study

In this section, we provide more ablation studies on the THUMOS14 dataset. Unless explicitly stated, we do not use the memory bank in these methods.

Attention normalization loss. In Table 1, we demonstrate the effectiveness of the attention normalization loss. As we can see, the attention normalization loss plays an important role in our method. It improves the performances of our baseline model as well as the full model. Besides, we can see that the full model without the attention normalization loss drops evidently, indicating that the attention normalization loss is essential to obtain useful representative snippets.

Representative snippet summarization. Here, we provide more experimental results to evaluate the representative snippet summarization. In our method, we generate the representative snippets regardless of the foreground or background. Since the classification head generates foreground scores, we can also force the model to only generate representative snippets of the foreground or background.

Table 2. The detection results of different variants of generating representative snippets.

Method	mAP @ IoU			
	0.3	0.5	0.7	AVG
Baseline	45.2	29.9	10.2	36.8
Full model w/o memory bank	54.5	37.3	12.5	44.2
Foreground representative snippet	52.2	33.2	11.7	41.8
background representative snippet	51.5	30.9	8.9	40.2
representative snippets of high scores	49.2	30.2	8.5	39.2
video-wise memory bank	56.2	38.2	12.4	45.0

Specifically, given the foreground scores $\mathbf{S}_f$ of the main branch, we first modulate the video snippet features $\mathbf{F}$ with $\mathbf{S}_f$ as $\mathbf{F}_f = \mathbf{F}\mathbf{S}_f$, and then use the EM attention to obtain the representative snippets based on the modulated features $\mathbf{F}_f$. We update the foreground features $\mathbf{F}_f$ by the bipartite random walk module, and finally summed up with the background features $\mathbf{F}_b = \mathbf{F}(1 - \mathbf{S}_f)$ to obtain the updated features. For the background representative features, we can follow the same pipeline to obtain the updated features.

In Table 2, we can see that, generating only foreground or background representative snippets cannot model the whole video, and thus the pseudo labels of background or foreground are still inaccurate. Besides, we also try to simultaneously generate foreground and background representative snippets by selecting the top-k and the bottom-k representative snippets according to their foreground scores. As we can see, this way obtains much inferior performance. Since different videos can be differently complex, manually selecting k foreground representative snippets and k background representative snippets is not enough to capture the variations of the whole video.

Memory bank. In Table 2, we also evaluate a variant of the memory bank. For each video, we use a separated memory bank to store all the representative snippets. During training, we randomly select one video of the same class from the dataset and retrieve its representative snippets as the offline representative snippets $\boldsymbol{\mu}^e$. We update the representative snippets in the memory bank by moving average just as the MoCo [4]. As we can see, this way achieves comparable performance with ours, indicating the effectiveness of propagating knowledge between videos. However, it requires much more memory (about $15\times$) to store the representative snippets.

Representative snippet propagation. Here, we provide more experimental results to evaluate the representative snippet propagation, the experimental results are shown in Table 3. We first evaluate the way of propagating the representative snippets, *i.e.*, the feature L1 loss and feature L2 loss in Table 3. Specifically, the feature L1 loss denotes that we enforce the original video features $\mathbf{F}$ and updated features $\mathbf{F}^a$ to be similar by L1 loss, *i.e.*, $\mathcal{L}_{l1} = \|\mathbf{F} - \mathbf{F}^a\|_1$. Likewise, the feature L2 loss denotes $\mathcal{L}_{l2} = \|\mathbf{F} - \mathbf{F}^a\|_2^2$. We can see that, both methods deteriorate the performance. This phenomenon may be due to the optimization objective

Table 3. The detection results of different variants of representative snippet propagation.

Method	mAP @ IoU			
	0.3	0.5	0.7	AVG
Feature L1 loss	52.1	32.1	10.1	41.1
Feature L2 loss	52.7	31.6	10.2	41.2
Concat $\boldsymbol{\mu}^a$ and $\boldsymbol{\mu}^e$	54.4	35.2	10.6	43.0

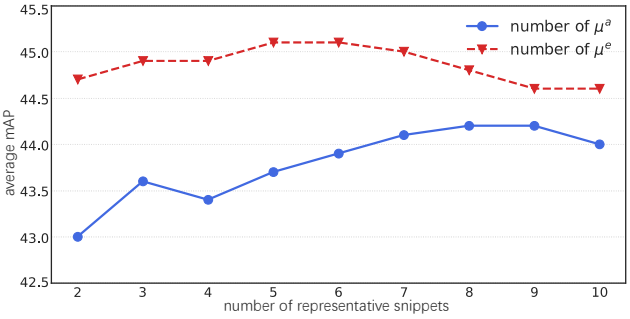

Figure 2. Ablation study on the numbers of representative snippets $\boldsymbol{\mu}^a$ and $\boldsymbol{\mu}^e$.

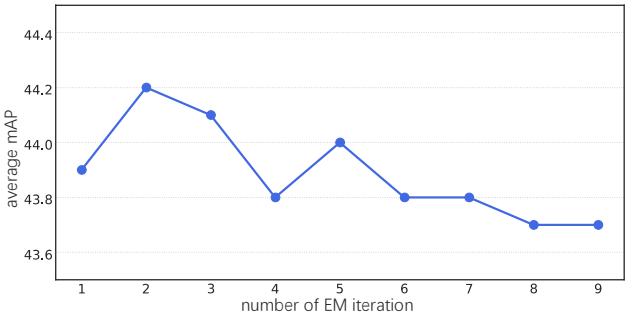

Figure 3. Ablation study on the number of EM iterations.

of the two loss functions is not the prediction scores but the features, which cannot directly guarantee consistent scores to obtain desirable detection results.

Moreover, in our method, we adopt the late fusion manner that combines the predictions of features updated with the online and offline representative snippets. Here, we also evaluate the early fusion strategy, which first concatenates the online and offline representative snippets and then propagate their knowledge to update the original video features and generate predictions. As we can see, this method obtains inferior performance, which may be due to the dominance of the online representative snippets in propagation.

Branch analysis. In our method, there are three branches, namely the main branch, the intra-video branch and the inter-video branch. In Table 4, we show their performance and MACs. First, we should highlight that inference can be done with only the main branch. Note that, all existing methods (including ours) use a fixed I3D (MACs 5500G). Except I3D's MACs, with only the main branch, our MACs (0.44G) is on par with W-TALC [11], but achieves 44.7% on

Table 4. Evaluation of three branches of their detection results and MACs. There are also the results of two methods W-TALC [11] and CO_2-Net [5] are demonstrated for comparison. All MACs are calculated with the same video having 100 frames and do not contain those of the fixed I3D.

Method	mAP @ IoU				MACs
	0.3	0.5	0.7	AVG	
W-TALC [11]	40.1	22.8	7.6	-	0.42G
CO_2-Net [5]	54.5	38.3	13.4	44.6	2.79G
Main branch	55.5	38.2	12.5	44.7	0.44G
+ intra-video branch	55.8	38.2	12.5	45.1	0.48G
+ inter-video branch	55.9	38.2	12.5	45.2	0.90G

THUMOS14, which is still higher than the best model CO_2-Net [5] (44.6%, MACs 2.79G) by 0.1% but with much less computational cost. Besides, snippet summarization, propagation and the intra-video branch incorporate little overhead, the main computation cost is the feature extraction module, which accounts for about 88% of the whole computation cost. Therefore, after adding the intra-video branch, our model (45.1%, MACs 0.48G) is still much more efficient than previous methods. Finally, we can see that further adding the inter-video branch can also improve the performance. However, since we do not know the video classes, we should first perform a round of inference to obtain the video prediction and then select the inter-video snippets according to the predicted video classes for the second round inference, which significantly increases the inference time. Therefore, we do not use the inter-video branch during testing in our final solution.

Number of representative snippets. In Figure 2, we evaluate the impact of the number of representative snippets on our method. When we evaluate the number of online representative snippets $\boldsymbol{\mu}^a$, we do not adopt the memory bank. When we evaluate the number of offline representative snippets $\boldsymbol{\mu}^e$, we fix the number of online representative snippet as the optimal value. We can see that our method can benefit much from the online representative snippets, even if we only utilize two Gaussians to model the foreground and background in the video, the performance is much higher than the baseline model (average mAP 36.8%). Our method achieves the optimal performance when the number of online representative snippets is 8. Moreover, our method is insensitive to the number of offline representative snippets, even 2 offline representative snippets can enable our method to achieve a promising performance. Our method achieves the optimal performance when the number of offline representative snippets of each class in the memory table is 5.

Number of EM iterations. In Figure 3, we evaluate the impact of the number of EM iterations on our method. As we can see, our method obtains promising representative snippets by 2 iterations. When the number of EM iterations increases, the performance slightly drop, which might be caused by gradient vanishing or explosion.

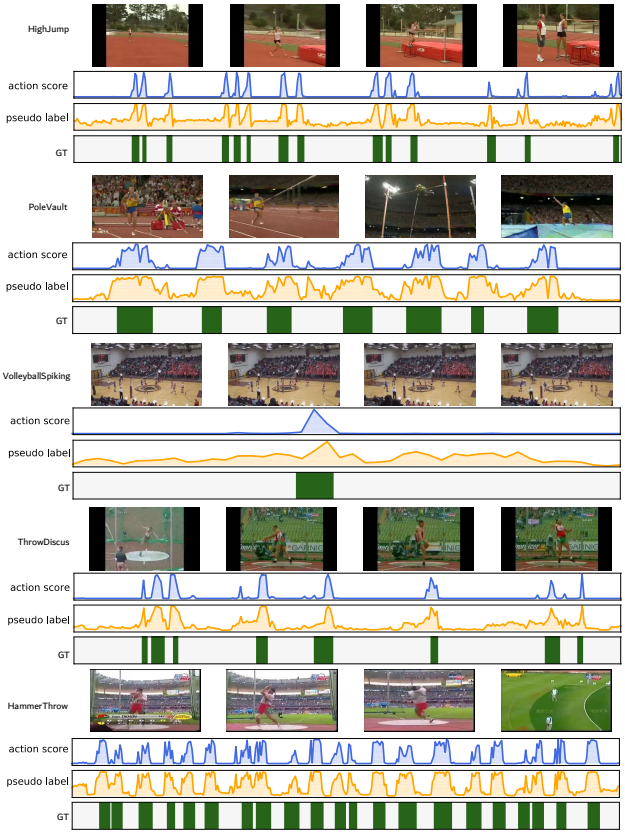

Figure 4. Qualitative results on THUMOS14 [7]. We show: 1) action activation scores, 2) pseudo label scores of ground truth action, 3) ground truth.

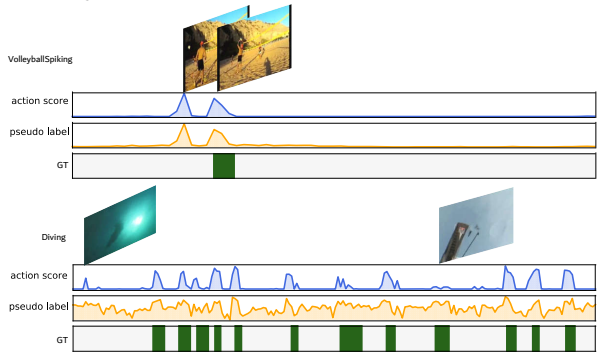

Figure 5. Some failure cases on THUMOS14 dataset. The two examples are *Volleyball Spiking* and *Cliff Diving*, respectively.

4.1. Qualitative Results

Detection results. We visualize some detection examples in Figure 4. We provide some frames of the input videos to show the corresponding actions. As we can see, for videos containing sparse or dense action instances, the localization results of our method are complete and accurate.

Failure cases. As shown in Figure 5, the first failure case of *Volleyball Spiking* is because the model confuses the preparing stage of this action, where the actors throw the

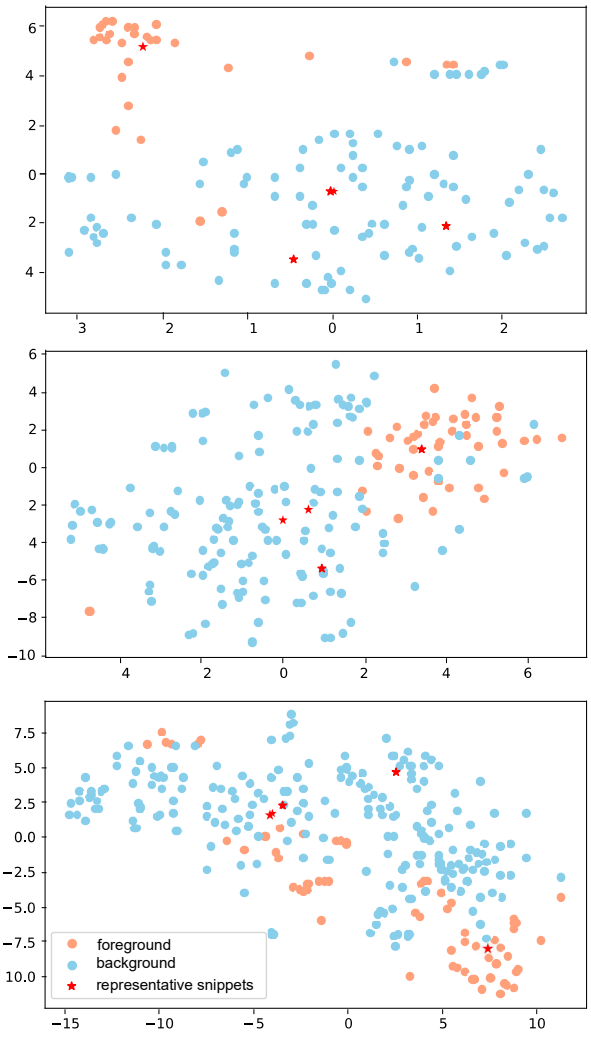

Figure 6. Visualization of the online representative snippets and the video features of the corresponding videos. The examples from top to bottom are *GolfSwing*, *CricketShot* and *Diving*, respectively.

volleyball before spiking. Due to the lack of frame-wise annotations, it is difficult to distinguish such fine-grained differences. The second failure case of *Cliff Diving* comes from the clustered background and small objects. As we can see, for the first false positive instance, there is a preview segment where an actor is diving into the water. It is hard to identify those snippets without strong supervisions. For the undetected instance, the actor is too small to provide enough appearance and motion cues.

Feature visualization. To attain further insights into the learned representative snippets, we visualize the online and offline representatives snippets and video features.

In Figure 6, we visualize the online representatives snippets and the video features of the corresponding videos. As we can see, the online representative snippets can well model the variations of the video, so as to describe most of the snippets of the same class.

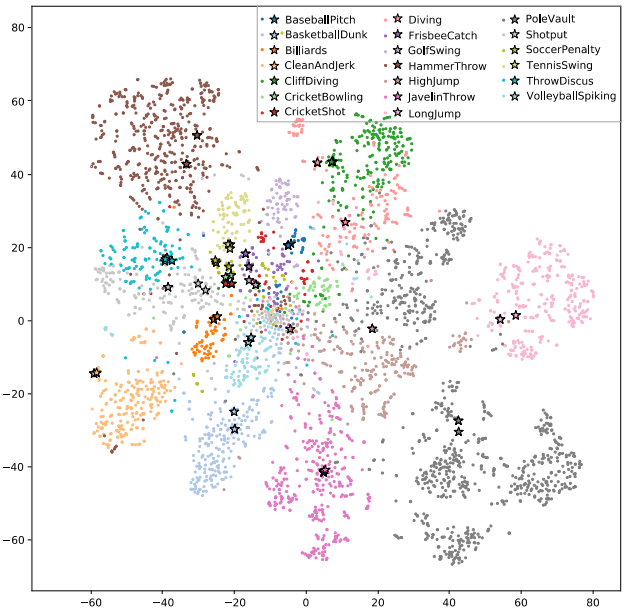

Figure 7. Visualization of the offline representative snippets and the video features of the dataset. Dots are the video features and stars are the offline representative snippets.

In Figure 7, we visualize the offline representatives snippets and the video features of the dataset. As we can see, the offline representative snippets usually locate at the dense area, demonstrating that they can represent the corresponding action class.

References

[1] Bahman Bahmani, Abdur Chowdhury, and Ashish Goel. Fast incremental and personalized pagerank. *arXiv preprint arXiv:1006.2880*, 2010. 1

[2] Gedas Bertasius, Lorenzo Torresani, Stella X Yu, and Jianbo Shi. Convolutional random walk networks for semantic image segmentation. In *CVPR*, pages 858–866. IEEE, 2017. 2

[3] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *CVPR*, pages 6299–6308. IEEE, 2017. 2

[4] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, pages 9729–9738. IEEE, 2020. 3

[5] Fa-Ting Hong, Jia-Chang Feng, Dan Xu, Ying Shan, and Wei-Shi Zheng. Cross-modal consensus network for weakly supervised temporal action localization. In *ACM MM*. ACM, 2021. 4

[6] Linjiang Huang, Liang Wang, and Hongsheng Li. Foreground-action consistency network for weakly supervised temporal action localization. In *ICCV*, pages 8002–8011. IEEE, 2021. 1

[7] Y.-G. Jiang, J. Liu, A. Roshan Zamir, G. Toderici, I. Laptev, M. Shah, and R. Sukthankar. THUMOS challenge: Ac-

tion recognition with a large number of classes. http://crcv.ucf.edu/THUMOS14/, 2014. 4

[8] Johannes Klicpera, Aleksandar Bojchevski, and Stephan Günnemann. Predict then propagate: Graph neural networks meet personalized pagerank. *arXiv preprint arXiv:1810.05997*, 2018. 1

[9] László Lovász. Random walks on graphs. *Combinatorics, Paul erdos is eighty*, 2(1-46):4, 1993. 1

[10] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab, 1999. 1

[11] Sujoy Paul, Sourya Roy, and Amit K Roy-Chowdhury. W-talc: Weakly-supervised temporal activity localization and classification. In *ECCV*, pages 563–579. Springer, 2018. 3, 4

[12] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 701–710, 2014. 1

[13] Yantao Shen, Hongsheng Li, Tong Xiao, Shuai Yi, Dapeng Chen, and Xiaogang Wang. Deep group-shuffling random walk for person re-identification. In *CVPR*, pages 2265–2274. IEEE, 2018. 2

[14] Yuanhao Zhai, Le Wang, Wei Tang, Qilin Zhang, Junsong Yuan, and Gang Hua. Two-stream consensus network for weakly-supervised temporal action localization. In *ECCV*, pages 37–54. Springer, 2020. 1