Interleaving One-Class and Weakly-Supervised Models with Adaptive Thresholding for Unsupervised Video Anomaly Detection

Yongwei Nie[1], Hao Huang[1], Chengjiang Long[2], Qing Zhang[3], Pradipta Maji[4], and Hongmin Cai[1]*

[1] South China University of Technology, China
[2] Meta Reality Labs, USA
[3] Sun Yat-sen University, China
[4] Indian Statistical Institute, India

Abstract. Video Anomaly Detection (VAD) has been extensively studied under the settings of One-Class Classification (OCC) and Weakly-Supervised learning (WS), which however both require laborious human-annotated normal/abnormal labels. In this paper, we study Unsupervised VAD (UVAD) that does not need any label by combining OCC and WS into a unified training framework. Specifically, we extend OCC to weighted OCC (wOCC) and propose a wOCC-WS interleaving training module, where the two models automatically generate pseudo-labels for each other. We face two challenges to make the combination effective: (1) Models' performance fluctuates occasionally during the training process due to the inevitable randomness of the pseudo labels. (2) Thresholds are needed to divide pseudo labels, making the training depend on the accuracy of user intervention. For the first problem, we propose to use wOCC requiring soft labels instead of OCC trained with hard zero/one labels, as soft labels exhibit high consistency throughout different training cycles while hard labels are prone to sudden changes. For the second problem, we repeat the interleaving training module multiple times, during which we propose an adaptive thresholding strategy that can progressively refine a rough threshold to a relatively optimal threshold, which reduces the influence of user interaction. A benefit of employing OCC and WS methods to compose a UVAD method is that we can incorporate the most recent OCC or WS model into our framework. Experiments demonstrate the effectiveness of the proposed UVAD framework. Our code is available at https://github.com/benedictstar/Joint-VAD.

Keywords: Video anomaly detection · One-class classification · Weakly-supervised learning · Unsupervised learning

1 Introduction

Video Anomaly Detection (VAD) is a task that identifies abnormal events in a video, where the abnormal event could be a fire alarm, a flaw in an indus-

* Corresponding author (ORCID: 0000-0002-2747-7234).

trial product, or a traffic accident, etc. Most previous VAD methods fall in two categories: One-Class Classification (OCC) methods [12, 13, 18, 19, 33, 39] and Weakly-Supervised (WS) approaches [21, 32, 34, 38, 43, 49]. However, both kinds of approaches require human-annotated labels for training. Note that OCC models are trained on normal data only, but it still requires human labor to exclude abnormal events in preparing the training dataset. For VAD, supervised methods have two prominent problems. First, abnormal events are sparse and difficult to recognize, making the annotation of a VAD training dataset a highly labor-intensive task. Second and more importantly, the categories of abnormal events are actually unbounded, while human can only collect a limited scope of them, yielding the risk that the trained VAD methods cannot handle unobserved abnormal events.

Considering the above problems, this paper studies Unsupervised VAD (UVAD) that does not rely on labels. Despite the importance of UVAD, few methods have been developed except [47] and its subsequent works [2, 37] to the best of our knowledge, probably because VAD is inherently a binary classification problem which naturally requires binary label's supervision. To achieve unsupervised learning, the method of [47] adversarially trains a generator and a discriminator, where the generator is an autoencoder-based reconstruction model and the discriminator is a fully-connected (FC) binary classifier. While the proposed network architecture is sophisticated, the adopted autoencoder and FC networks are relatively weak which limit the performance of the approach.

In this paper, we propose a new UVAD framework. Our key idea is that the techniques of OCC and WS approaches are rapidly developed, while a UVAD method is usually implemented by training two VAD models alternately generating pseudo labels for each other. This motivates us to propose a UVAD framework that directly interleaves the training of a pair of OCC and WS models. In this way, we are flexible to incorporate the recent advances in the two hot research fields to implement a UVAD method. Note that we have also attempted to combine two OCC or two WS models, but found two homogeneous models are less effective.

While combining OCC and WS methods, we encounter two problems preventing it from effective. First, we observe the OCC or WS model's performance fluctuates occasionally during the training process, which affects the final accuracy they can achieve. This is partly because the pseudo labels generated by each other are changed frequently, making the training not stable. To solve this problem, we extend OCC to weighted OCC (wOCC), and propose a wOCC-WS interleaving training module. Previous OCC model is trained on normal data, while our wOCC model is trained with soft labels in the range of $[0, 1]$. The soft labels are more consistent across adjacent training cycles, making the performance of wOCC more stable than OCC under changing pseudo labels, thus suppressing the fluctuation effect. Second, we still have to set a threshold to partition normal and abnormal pseudo labels for the WS model, making our method susceptible to a user-provided hyperparameter. To reduce the influence of user interaction, we repeat the interleaving training module multiple times,

during which we propose an adaptive threshold mechanism that finds an optimal threshold given just a rough initial value from the user.

To conclude, the contributions of this work include:

- We propose a new UVAD framework that interleaves the training of a pair of OCC and WS models.
- We improve OCC to wOCC which stabilizes the training of the UVAD framework, and also propose an adaptive thresholding strategy to alleviate the influence of user interaction.
- Our UVAD framework is flexible to employ different pairs of OCC and WS models. Experiments on three OCC and two WS models demonstrate the effectiveness of our method.

2 Related Work

Unsupervised VAD (UVAD). Unsupervised VAD requires identifying anomalies from data containing both normal and abnormal samples without any annotation. This is a challenging new task that is almost unexplored in the literature. Although significant progress has been made in OCC and WS VAD, directly applying them independently to address UVAD does not yield good results. Zaheer et al. [47] firstly raise this task and propose to solve UVAD by generative cooperative learning (GCL). Differently, our proposed method modifies both the OCC and WS VAD models and combines the modified two models together to learn from unmarked training data.

One-Class Classification VAD (OCC). OCC-based VAD approaches only have access to normal data and try to model normal data to identify behaviors that are significantly different from normal behaviors as anomalies. Early works use hand-crafted features to help detect anomalies [5, 23, 40, 51]. With the rapid development of deep learning, recent methods turn to normal representations extracted by using a deep neural network [26, 30, 36, 41]. Some methods identify normal patterns by using the reconstruction/prediction model to reconstruct the representations [6,12,14,18,19,24,25,27,31,33,45,46,52]. These models may lead to well-reconstructed anomalies, thereby limiting the performance of the OCC-based methods. Other OCC-based methods turn to identify normal representations by using proxy tasks [4, 11, 20, 29, 39]. What's more, recently, Hirschorn and Avidan [13] propose to build a multivariate Gaussian distribution for normal data and detect instances deviating from this distribution as anomalies. However, the above methods treat all samples equally, ignoring the potential differences in importance between normal samples.

Weakly-Supervised VAD (WS). In weakly-supervised VAD, video-level annotations are available in the training stage. Sultani et al [32] first propose to use the video-level labels and solve WS VAD based on multi-instance learning (MIL) framework [3, 7]. Since then, many works [21, 28, 38, 43, 49, 50] have viewed the WS VAD task as a MIL problem. Tian et al. [34] develop to extend MIL to Top-k MIL method by training with a robust temporal feature magnitude

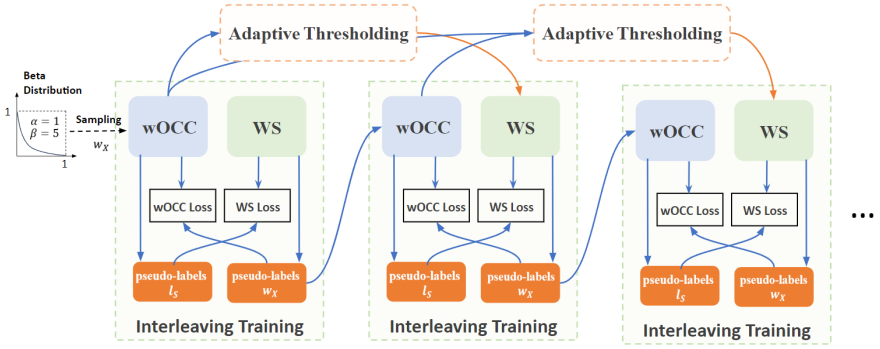

Fig. 1: Pipeline of our proposed interleaved framework. The wOCC-WS interleaving training module comprises a weighted OCC (wOCC) model and a WS model, using the output of each model as the pseudo-labels to supervise the other model. Further, the training module is repeated multiple times with the adaptive threshold until the stopping criterion we set. To start the first training module, we randomly sample w_X to train the wOCC model and assign a large value as the threshold for the WS model. Once a training module converges, the last WS model of it supplies w_X to start the next training module and adaptive thresholding takes the outputs of all previous wOCC models into consideration for updating the threshold.

(RTFM) loss function. The methods mentioned above directly train in the supervision of video-level labels and the coarse-grained supervision limits the accuracy of WS models. Recently, two-stage self-training methods [8, 17, 44, 53] adopt a two-stage pipeline to use more fine-grained labels to supervise the networks more strictly. The WS model in our framework shares a similar idea with [8] and is tailored to use finer-grained snippet-level labels.

3 Method

3.1 Overview

Our proposed UVAD framework interleaves the training of a pair of OCC and WS models. In theory, any OCC or WS methods can be incorporated into our framework. Without loss of generality, we take STG-NF [13] (a recent OCC model) and RTFM [34] (a recent WS model) as examples to introduce our method in the following sections. We test many other OCC/WS models in Section 4.5.

Figure 1 shows an overview of our framework. The basic buildingblock of our method is the wOCC-WS interleaving training module. In the module, the outputs of each model are used as pseudo labels for defining the loss function training the other model. After one module converges, we re-initialize another module to run the interleaving training again with a difference that the threshold of the WS model is updated by an adaptive threshold adjustment mechanism based on wOCC models trained in previous modules. Since wOCC has more stable behaviors than WS, we always initialize the wOCC model to start the

training of a module. For the first module, we use a random initialization based on Beta distribution sampling. For other modules, we use the results of WS of the preceding module for the initialization.

3.2 wOCC-WS Interleaving Training Module

We first introduce the wOCC-WS interleaving training module which comprises a wOCC model and a WS model. At the very beginning, we initialize the wOCC model to start the interleaving training. Then, pseudo labels created using wOCC are used to train the WS model. Next, pseudo labels created by the WS model are used to improve the wOCC model. The whole training does not use human annotations.

wOCC Model (STG-NF). Typically, OCC models are trained only on normal data. Directly using OCC in our UVAD framework may make the training fluctuate, i.e., the performance of the trained models sometimes gets better, but sometimes worse, finally converging to an inferior point. We use 0 to indicate normal samples, and 1 abnormal samples, where 0, 1 are hard binary labels. During training, we need to identify 0-labelled samples to train the OCC model. The hard partitioned labels may be suddenly changed from 0 to 1 or from 1 to 0, yielding the frequent change of the normal samples used to train the OCC model which is the source of the unstable training. To this end, we propose wOCC which relies on soft labels instead of hard binary labels. Soft labels are in the range of $[0, 1]$. For example, the soft label 0.7 may be changed to 0.6 but not 0 suddenly. This consistency reduces the fluctuation of the interleaving training.

Next, taking STG-NF [13] (an OCC model) as an example, we introduce how to improve it to wOCC. STG-NF extracts a set of objects $\mathcal{X} = \{X_1, \cdots, X_i, \cdots\}$ from the training videos, where X_i is a sequence of poses of a person. Each object X_i is the basic building block processed by STG-NF. Originally, STG-NF minimizes the negative log-likelihood across the normal data:

$$\mathcal{L}_{occ} = -log(p_Z(f_{STG-NF}(X_i^+))), \tag{1}$$

where $X_i^+ \in \mathcal{X}^+$ is sampled from normal data, $Z = f_{STG-NF}(X_i^+)$ is the feature extracted from X_i^+ by the STG-NF, p_Z is the established distribution of the normal data. We upgrade STG-NF by introducing the soft labels $w_X = \{w_{X_1}, \cdots, w_{X_i}, \cdots\}$ as the weight to supervise its training on both normal and abnormal data. Formally, for the wOCC model improved from STG-NF, we minimize the following weighted negative log-likelihood across all the data :

$$\mathcal{L}_{wocc} = -(1 - w_{X_i})log(p_Z(f_{STG-NF}(X_i))), \tag{2}$$

where $X_i \in \mathcal{X}$ is sampled from the whole training dataset rather than the normal part and $w_{X_i} \in [0, 1]$ is the weight of the object X_i indicating the anomaly degree of the object which serves as the soft label. The object with smaller w_{X_i} means more normal which is better captured by the learned distribution of wOCC.

Pseudo Labels l_S from wOCC. Now we use the trained wOCC model to generate pseudo labels $l_S = \{l_{S_1}, \cdots, l_{S_i}, \cdots\}$ for training the WS model.

"Snippet" is the basic element processed by WS models. We first split the training videos into snippets, obtaining a set of snippets $\mathcal{S} = \{S_1, \cdots, S_i, \cdots\}$. Our aim is to compute the label $l_{S_i} \in \{0, 1\}$ for each snippet S_i. Let x_i be the anomaly score of the object X_i computed by the trained wOCC model. The anomaly score $\hat{s}_i$ of a video snippet S_i is computed by averaging all the anomaly scores of objects in the snippet. To generate pseudo label l_{S_i}, one way is to check whether $\hat{s}_i$ is larger than a threshold ($l_{S_i} = 1$) or not ($l_{S_i} = 0$). However, as the training goes, the anomaly scores of snippets may vary in a large range, making the absolute threshold invalid. Instead, we sort the snippets according to their anomaly scores in descending order, and get the index of the snippet S_i by $\text{Rank}(\hat{s}_i)$ in the sorted list. Finally, the label l_{S_i} of snippet S_i is computed by

$$l_{S_i} = \begin{cases} 1, & \text{if} \quad \text{Rank}(\hat{s}_i) < T_{ws}, \\ 0, & \text{otherwise}, \end{cases} \tag{3}$$

where T_{ws} is the relative threshold not affected by concrete anomaly scores.

WS Model (RTFM). With snippet-level pseudo annotations $l_{\mathcal{S}}$ (obtained in Eq. 3), we now define positive and negative bags and use them to train a WS model. Previously, approaches [32,34] usually treat snippets of the same video as a bag based on video-level labels. Differently, thanks to the snippet-level labels provided by the wOCC model, we can compose positive and negative bags more flexibly. In our method, a positive bag $B^+ = \{S_i^+\}_{i=1}^C$ is composed of randomly selected $C = 16$ abnormal snippets. Similarly, a negative bag $B^- = \{S_j^-\}_{j=1}^C$ is composed of randomly selected C normal snippets. Taking RTFM [34] as an example, the WS method employs multiple instance learning, requiring the average Top-k maximum magnitudes of features in positive bags exceeds that in negative bags:

$$\begin{aligned} \mathcal{L}_{ws} = \max(0, m - d_{RTFM}(B^+) + d_{RTFM}(B^-)) + \\ \mathcal{L}_{BCE}(B^+) + \mathcal{L}_{BCE}(B^-), \end{aligned} \tag{4}$$

where m is the margin, and $d_{RTFM}(B)$ (where B can be B^+ or B^-) returns the average of the Top-k maximum magnitudes of features. Finally, $\mathcal{L}_{BCE}$ is the binary cross-entropy-based classification loss:

$$\mathcal{L}_{BCE}(B) = -Y \log(f_{RTFM}(B)) + (1 - Y) \log(1 - f_{RTFM}(B)), \tag{5}$$

where Y is the label of the bag B and $f_{RTFM}(B)$ returns the average of the Top-k maximum anomaly scores. Note that $f_{RTFM}(B)$ is different from $d_{RTFM}(B)$, where $f_{RTFM}(B)$ returns the anomaly scores while $d_{RTFM}(B)$ returns the magnitude of the features.

Pseudo Labels w_X from WS. wOCC model needs soft labels w_X. With the trained WS model, we compute the anomaly score $\hat{x}_i$ of object X_i by simply setting it as the anomaly score of the snippet containing the object X_i. Then, the soft label w_{X_i} is directly set as:

$$w_{X_i} = \hat{x}_i. \tag{6}$$

$\hat{x}_i$ is already in the range of $[0, 1]$, so we do not need to normalize w_{X_i}.

Initializing the Interleaving Training Module. As discussed above, in our proposed interleaving training module, wOCC training relies on pseudo labels generated by the WS model, and conversely, WS training requires pseudo labels derived from the wOCC. This egg and chicken problem needs to be solved at the very beginning. Our solution is to provide the wOCC model with randomly generated soft labels to start the interleaving training. Usually, the training dataset contains much more normal samples than abnormal ones, which means most soft labels are around 0 while a small portion of them are near 1. Therefore, we randomly sample the weight w_X from the Beta distribution, i.e., $w_X \sim \text{Beta}(\alpha, \beta)$, where $\alpha = 1$ and $\beta = 5$. The sampled weights w_X are mostly around 0, and a small portion of them is close to 1, satisfying the prior about the distribution of normal and abnormal data.

Convergence Analysis. We have discussed the mechanism and the way to start the interleaving training module above. Then, in the following, we empirically analyze its convergence, which is validated by the experiment in Figure 6. At the very beginning, we randomly initialize the weight w_X to train our wOCC model. Although the random initialization is not precise, the wOCC model can still learn normal patterns from the training data, as we assume there is more normal data than abnormal data in the training dataset. Therefore, the wOCC model can produce relatively reliable pseudo labels for the WS model. This is very important as the WS model has a high requirement on the quality of the pseudo labels. In turn, with the trained WS model, we can provide better pseudo supervision to train a stronger wOCC model than the random initialization. In this way, we achieve mutual improvement between two models until convergence.

The above analysis implies the reason why we do not choose to interleave the training of two OCC models or two WS models. The OCC model can provide reliable initialization, while the WS model explicitly enlarges the gap between normal and abnormal samples which better exploits the utilization of both the normal and abnormal data. Two OCC models fail to utilize the abnormal data. For two WS models, it is not known how to provide a reliable initialization to start the training.

3.3 Repeating Procedure with Adaptive Thresholding

In our proposed interleaving training module, the wOCC and WS models need to provide pseudo labels to supervise the other. As the wOCC model relies on soft labels w_X as defined in Eq. 6, its pseudo labels can be directly provided by the WS model without thresholding. However, as defined in Eq. 3, we need to specify a threshold for the WS model since it requires binary hard partition labels to construct positive and negative bags. To alleviate the influence of human interaction, we propose a repeating procedure of the interleaving training module which can adaptively refine a user-specified rough threshold to a relatively optimal one.

Starting and running the next interleaving training module. As shown in Figure 1, we run several interleaving training modules one after another.

Recall that we provide w_X by random initialization to start the first interleaving training module as discussed in Section 3.2. To start the next module, we use the final WS model in the last module to provide the labels w_X. Note that for the wOCC and WS models in a new interleaving training module, we train them from scratch, not inheriting parameters from the last module.

Adaptive Thresholding. The threshold is fixed in each module. Between two modules, we employ an adaptive thresholding mechanism to refine the threshold. As the repeating procedure runs, we obtain a series of thresholds $\{T_{ws}^1, T_{ws}^2, \cdots, T_{ws}^i, \cdots\}$ corresponding to the interleaving training modules. The key idea of adaptive thresholding is to ensure:

$$T_{ws}^1 \geq T_{ws}^2 \geq \cdots \geq T_{ws}^i \geq \cdots . \tag{7}$$

Starting from a sufficiently large (larger than the optimal threshold) value of T_{ws}^1, there must be one threshold $T_{ws}^* \in \{T_{ws}^1, T_{ws}^2, \cdots, T_{ws}^i, \cdots\}$ that is closest to the optimal threshold. The problem is how to enforce the monotonically decreasing of the threshold and how to stop the repeating procedure at the optimal point.

First, we initialize $T_{ws}^1 = R\% \times N$, where R is a user-specific hyper-parameter, e.g. $R = 30$, and N is the total number of snippets in the training dataset. This setting makes T_{ws}^1 a large enough threshold, since generally the ratio of abnormal data in the training dataset is less than 30%. We test different R (e.g., 35%) in the experimental section and find that it does not influence our final results much, demonstrating the robustness of our method to the user interaction.

Then, to ensure Eq. 7, our adaptive thresholding method determines the threshold used in the current module based on wOCC models trained in all previous modules. Each training module produces multiple wOCC models, obtaining one after each loop between wOCC and WS. Let M_i be the number of wOCC models trained from module 1 to module i. When computing T_{ws}^{i+1} for module $i+1$ with $i \geq 1$, we take all the M_i wOCC models into consideration. Specifically, for each wOCC, e.g., the j^{th} one where $j \in [1, M_i]$, we use it to identify $R\%$ of snippets in the training dataset that have high anomaly scores, and denote the resulting set of snippets as A_j. We then compute intersection between all sets and count the number of the resulting intersection as the threshold:

$$T_{ws}^{i+1} = \text{Num}(A_1 \cap A_2 \cap \cdots \cap A_{M_i}), \tag{8}$$

where $\text{Num}(\cdot)$ counts the number of elements in the intersection. The operation of computing intersection ensures that the obtained threshold is monotonically decreased. In essence, the threshold computed in this way means the number of snippets viewed as abnormal by all wOCC models. At the early stages, the number of wOCC models is small, therefore the size of the intersection is large, yielding large thresholds. As more wOCC models involved, it is more difficult to form a consensus, thus producing smaller thresholds.

To elucidate the effect of the adaptive thresholding, we visualize how WS models are gradually improved as the repeating procedure runs in Figure 2. In the figure, we show two examples. Taking the one on the left as an example, it shows a video with an abnormal event in the middle. At the beginning when

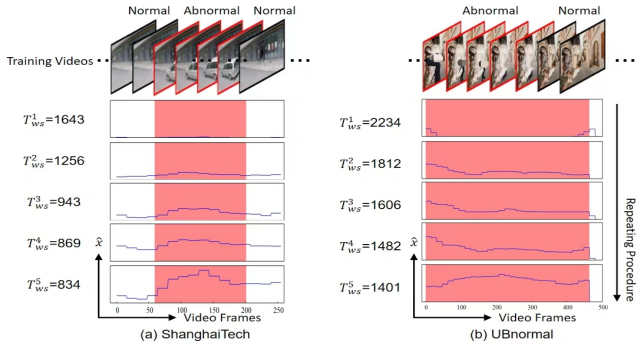

Fig. 2: As the repeating procedure goes, T_{ws} becomes smaller and smaller, while the WS model is improved gradually. Left: an example from ShanghaiTech dataset. Right: an example from UBnormal dataset.

$T_{ws} = 1643$, the WS model wrongly identifies the abnormal event as normal. As the repeating procedure runs, T_{ws} becomes smaller and smaller, which is finally reduced to 834 after repeating 5 times, and the WS model begins to identify the anomaly event progressively.

Stopping Criterion. As our goal is to find the threshold T^*_{ws} close to the optimal threshold, we need to stop the repeating procedure at the right time. Experimentally, we find that T_{ws} drops fast at the first few interleaving training modules, then the rate of change of T_{ws} becomes smaller which indicates that the wOCCs achieve a consensus about anomalies. The size of the consensus set is probably the actual number of abnormal snippets in the dataset. Based on this observation, once change rate of T_{ws} between two consecutive training modules is less than $Q\%$ of the first change rate at the very beginning, we stop the repeating procedure. Please see the high correlation between the change rate of T_{ws} and the accuracy of the WS model in Figure 4.

Training Time Analysis. Though we need to repeat the interleaving training module multiple times, our training is as fast as the original wOCC or WS method. First, the repeating procedure usually stops with fewer than 6 modules thanks to the stopping criterion. Furthermore, we train wOCC (or WS) for just one epoch, and then exchange to train WS (or wOCC) for another epoch, which greatly saves the training time. If we have 6 modules, each running an average of 5 loops of interleaving training, the wOCC and WS models are trained for just 30 epochs each. The original wOCC or WS models are also trained for the similar number of epochs in their default experimental settings. The detailed experiments are in Section 4.5.

3.4 Inference

At the inference stage, both wOCC and WS models can be used to detect anomalies. We provide results of both models when compared with previous methods.

4 Experiments

4.1 Datasets and Evaluation Metrics[1]

ShanghaiTech. The ShanghaiTech dataset [18] contains 437 videos and was created as a benchmark for OCC with only normal videos available in the training set. Zhong et al. [53] reorganized the dataset to enable the training of WS systems. The new split contains 63 abnormal videos and 175 normal videos, while there are 44 anomalous and 155 normal videos in the new testing split. In our unsupervised setting, we follow the split for WS approaches but do not provide video-level annotations in the training stage.

UBnormal. The UBnormal dataset [1] is a synthetic open-set benchmark, containing 268 training videos, 64 validation videos and 211 test videos. Normal and abnormal videos are mixed in the three subsets. The dataset is challenging because of disjoint sets of anomaly types in training and testing sets. We follow its original organization but train our proposed UVAD without any annotations.

Evaluation Metrics. Following prior VAD arts [13, 34], we use the frame-level area under ROC curve (AUC) to measure VAD's accuracy for both datasets.

4.2 wOCC and WS models Employed

As stated, our method is a flexible UVAD framework that can use different wOCC and WS models. We test improving three OCC models to wOCC, including the AE model used in [47], Jigsaw [39], and STG-NF [13]. *The wOCC losses for AE and Jigsaw are put into supplemental material.*

(1) **AE.** GCL [47] proposes to use an AutoEncoder that reconstructs the features extracted from videos as their OCC model. We use it here too.

(2) **Jigsaw.** The OCC model Jigsaw [39] addresses VAD by solving a pretext task: spatio-temporal jigsaw puzzles. It divides the spatio-temporal space of a video into smaller cubes, then shuffles the positions of the cubes, and finally tries to restore the original positions of the cubes.

(3) **STG-NF.** STG-NF [13] extracts people's pose sequences, and extends Glow [16] to build multivariate Gaussian distribution of normal action sequences.

For WS, we test the method proposed in Sultani et al. [32] and RTFM [34].

(1) **Sultani et al.** [32]. Sultani et al. [32] propose the first MIL VAD model that maximizes the separability between a positive bag containing snippets of an abnormal video and a negative bag with snippets of a normal video.

(2) **RTFM.** RTFM [34] extends Sultani et al. [32] to compare the Top-k largest-magnitude snippets between positive and negative bags.

[1] The authors Hao Huang and Yongwei Nie signed the license and produced all the experimental results in this paper. Meta and Meta employees did not have access to the datasets.

Table 1: Comparison with previous approaches. For wOCC, we use STG-NF [13]. For WS, we use RTFM [34]. We test using C3D [35] and I3D [15] to extract features for RTFM. Our$_{wOCC}$ gives the AUC of the wOCC model. Our$_{WS}$ gives the AUC of the WS model. The methods split by the dashed line are trained as a whole in our framework. * means we reimplement the method by taking I3D as the feature. ** means our method degenerated to the supervised OCC or WS method. *** means the results on UBnormal are reproduced by us using the official code.

Unsupervised			
Method	**Features**	**STech AUC %**	**UB AUC %**
AE$_{AllData}$	I3D	60.51	-
STG-NF$_{AllData}$ [13]	-	80.29	70.48
GCL* [47]	I3D	76.14	-
Our$_{wOCC}$	-	81.50	71.67
Our$_{WS}$	C3D	85.43	59.05
Our$_{wOCC}$	-	82.57	**74.76**
Our$_{WS}$	I3D	**88.18**	63.10

One-Class Classification		
Method	**STech AUC %**	**UB AUC %**
MemAE [12]	71.20	-
Frame Prediction [18]	73.40	-
Markovitz et al. [22]	76.10	52.00
HF^2-VAD [19]	76.20	-
CT-D2GAN [9]	77.70	-
CAC [42]	79.30	-
SSMTL [10]	82.40	-
Georgescu et al. [11]	82.70	59.30
SSMTL++ [4]	83.80	62.10
Jigsaw [39]	84.30	56.40
STG-NF [13]	85.90	71.80
Our$^{**}_{wOCC}$ (STG-NF)	**86.37**	**72.81**

Weakly Supervised			
Method	**Features**	**STech AUC %**	**UB AUC %**
GCN [53]	C3D	76.44	-
GCN [53]	TSN	84.44	-
Zhang et al. [50]	C3D	82.50	-
Sultani et al.*** [32]	I3D	84.53	54.12
AR-Net [38]	I3D	85.38	-
CLAWS [48]	C3D	89.67	-
MIST [8]	I3D	94.83	-
Li et al. [17]	I3D	96.08	-
RTFM*** [34]	C3D	91.51	62.30
RTFM*** [34]	I3D	97.21	66.83
S3R [43]	I3D	97.48	-
Our$^{**}_{WS}$ (RTFM)	I3D	**97.53**	**67.42**

Table 2: Ablation study on wOCC improved from different OCC models, always with RTFM as the WS model.

OCC Model	Our$_{wOCC}$	Our$_{WS}(RTFM)$	GCL$_{Classifier}$
AE	70.99	78.90	76.14
Jigsaw [39]	81.23	85.35	-
STG-NF [13]	82.57	88.18	-

Table 3: Ablation study on using different WS models in our method, always with STG-NF as the wOCC model.

WS Model	Our$_{wOCC}$ (STG-NF)	Our$_{WS}$
Sultani et al. [32]	81.92	77.41
RTFM [13]	82.57	88.18

4.3 Implementation Details

We implement our method in PyTorch. For STG-NF and RTFM (implementation details of other wOCC/WS models are put in supplemental material), they are optimized by Adam optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.999$, and a weight decay of 0.0005. The batch size of STG-NF is 256, and that of RTFM is 32. The learning rate is $5e-4$ and $1e-3$ for STG-NF and RTFM respectively. For RTFM, the margin m is set to 100. During interleaving training, we train a model for one epoch and then exchange to train the other model for another epoch.

Our method has two hyperparameters shared by all types of wOCC or WS models. One is $R\%$ which determines the initial threshold T^1_{ws}. By default, we set it to 15% for the ShanghaiTech dataset which contains a small ratio of abnormal data. The UBnormal dataset contains relatively more abnormal data and we set $R\%$ to 30%. The other parameter is $Q\%$ used to stop the repeating procedure. For both datasets, we set $Q\%$ to 10%. Analysis of Q% is put into *Supp.*

4.4 Comparison with Previous Approaches

From left to right, Table 1 shows the comparison between our and previous approaches under Unsupervised, One-Class Classification and Weakly-Supervised. Since the unsupervised methods are scarce, besides GCL [47], we compare with

AE$_{AllData}$ which means training the AE model on the whole training dataset containing both normal and abnormal data. We also compare with STG-NF$_{AllData}$. For our method, we adopt STG-NF [13] as wOCC, and RTFM [34] as WS.

First, Our$_{wOCC}$ (trained along with RTFM (I3D)) outperforms STG-NF$_{AllData}$ (see the table on the left). This validates the benefit of using our method to distinguish the normal from abnormal data in the training dataset. On ShanghaiTech, our proposed wOCC improves STG-NF from 80.29% to 82.57%. In contrast, the improvement is from 70.48% to 74.76% on UBnormal. The improvement on UBnormal is larger as there is more abnormal data in UBnormal which causes STG-NF$_{AllData}$ to perform worse.

Second, please compare our method with GCL. Our$_{WS}$ achieves an AUC of 88.18%, while that of GCL is 76.14% (this score is reported by the classifier of GCL, thus we use our WS model for the comparison). This significant improvement is partly due to the reason that we use a stronger OCC model, *i.e.*, STG-NF, than GCL. If using the same OCC model, our method outperforms GCL by 2.76%, as shown in Table 2.

Our UVAD method can be degenerated into a supervised OCC method. For example, we can perform our interleaving training on the normal data only. The obtained wOCC model in this setting is denoted by Our$^{**}_{wOCC}$ which outperforms existing supervised OCC approaches (see the table in the middle). This validates the effectiveness of the proposed wOCC with weighted importance, as even among the normal data there is data that is more normal than other data, and the wOCC model can better identify their difference.

To degenerate our method into a supervised WS method, we allow our WS model to know video-level labels. Specifically, snippets of normal videos are treated as normal. Since snippets of abnormal videos can be either normal or abnormal, they are processed by our adaptive thresholding approach. As seen in Table 1 on the right, Our$^{**}_{WS}$ (RTFM) outperforms the baseline RTFM [34]. On UBnormal, our method performs best among all the compared WS methods.

4.5 Ablation Study and Discussions

Next, we perform ablation studies on our method. Without extra explanations, experiments are conducted by taking STG-NF as wOCC and RTFM as WS.

Comparison between OCC over wOCC. In Figure 3, we compare the performance fluctuation of using wOCC or OCC (STG-NF) in our framework in the first two interleaving training modules' training. As shown, directly using OCC leads to performance fluctuation of both OCC and WS models and converges to an unsatisfactory point. In comparison, our proposed wOCC stabilizes the training and improves the performance of both models.

Using Different OCC/WS Models. In Table 2, we conduct experiments of upgrading different types of OCC models to wOCC in our method. We test AE [47], Jigsaw [39], and STG-NF [13], while always using RTFM [34] as the WS model. As seen, the effectiveness of our method is essentially affected by wOCC model. Similarly in Table 3, the same wOCC model is used but with different WS models. Overall, a better wOCC or WS model yields better results.

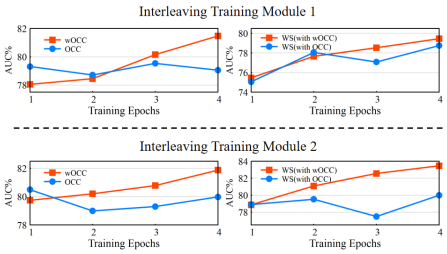

Fig. 3: Top: performance fluctuation in the first module. Bottom: performance fluctuation in the second module. Red line: AUC of wOCC (or WS) when interleaving wOCC and WS. Blue line: AUC of OCC (or WS) when interleaving OCC and WS.

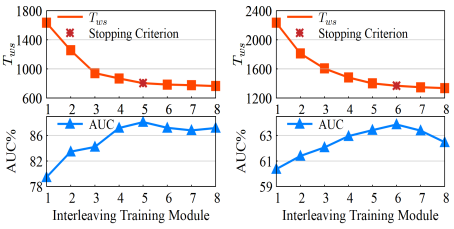

Fig. 4: Left: ShanghaiTech. Right: UBnormal. Top: T_{ws} in each interleaving training module. Bottom: The AUC of the WS model in each module. Stars in top figures: position where achieves stopping criterion.

Table 4: Comparison with fixed thresholding.

Weighted OCC	WS with Adaptive Threshold	RTFM (AUC %)	STG-NF (AUC %)
✗	✗	82.06	80.52
✓	✗	83.48	81.78
✗	✓	85.86	81.94
✓	✓	**88.18**	**82.57**

Comparison with Fixed Thresholding. In Table 4, the first row shows results of using fixed thresholding for both OCC and WS models. The second and third rows use fixed thresholding for either OCC or WS models. The fourth row shows our method with wOCC and adaptive thresholding. As seen, our method outperforms all the other variants.

Effectiveness of the Stopping Criterion. In Figure 4, we show that with the current stopping criterion, our method can stop when it achieves the best AUC (*i.e.*, the best VAD accuracy). As the repeating procedure runs, T_{ws} drops and AUC increases. T_{ws} drops very fast at the very beginning, and then the drop rate becomes slower. Simultaneously, AUC first increases and then decreases. We stop our method when the change rate of T_{ws} is about 10% of its original change rate, which finds the best stopping positions indicated by the stars in top figures.

Robustness to R%. Our method is robust to the parameter R%, as demonstrated in Figure 5. On the top, we show that the threshold T_{ws} converges to similar values after 6 interleaving training modules when using different R%. At the bottom, we show that different R% yield similar AUC on both training datasets. The experiments demonstrate that our method is not influenced much by this parameter for each dataset. For different datasets, we need to set different R% that roughly match with the ratio of anomaly data in the datasets.

Training Loss Curve. In Figure 6, we show training loss curves of the wOCC (STG-NF) model in our framework. As seen, the loss drops smoothly within each interleaving training module. Since we re-train the wOCC model from scratch in the next module, the loss suddenly increases at the beginning of

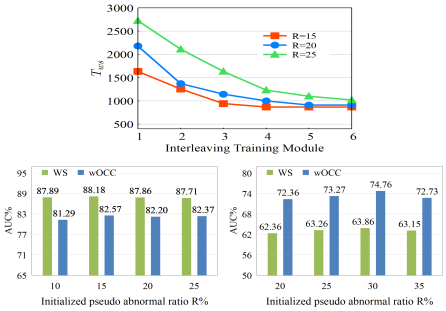

Fig. 5: Left: different $R\%$ converge to similar T_{ws} (on STech). Middle: different $R\%$ yield similar AUC (on STech). Right: different $R\%$ yield similar AUC (on UB).

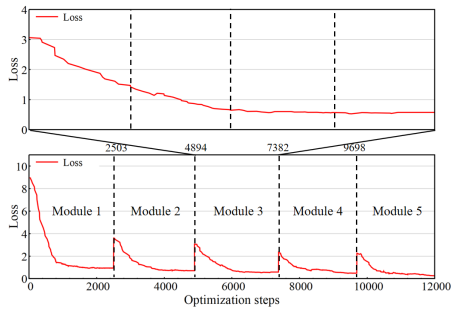

Fig. 6: Bottom: training loss curve of the wOCC model during the whole repeating procedure. Top: zoomed-in training loss curve in the third module.

Table 5: Training time on ShanghaiTech.

Model	Epoch Each Step	Training Steps	Training Epochs	Training Time	STG-NF AUC %	RTFM AUC %
Our UVAD	1	17	2×1×17	2h28m	**82.57**	**88.18**
	2	12	2×2×12	2h42m	82.53	87.95
	3	11	2×3×11	3h31m	82.27	86.36
STG-NF [13]	-	-	8	10m	85.90	-
RTFM [34]	-	-	50	2h32m	-	97.21

the next module, but the peak magnitude of the loss is lower than that of the previous module. Please check the zoomed-in curve showing the loss curve of the wOCC model in the third module. Even though the model is trained alternately with another model, its loss can still decrease smoothly, not affected by the other model. The WS model's training loss curve is put into the supplemental material.

Training Time. As mentioned in Section 3.3, although we need to train the wOCC and WS models alternately many times in multiple modules, our method is fast in training. That is because, we just train a model for one epoch and then exchange to train the other model for another epoch, which form a training step of our method. Our interleaving training converges very fast. As shown in Table 5, there are only 17 training steps during the whole training, yielding a total of 34=2(wOCC and WS)×1(epoch)×17(all training steps) epochs. It takes around 2.5 hours to train our method on ShanghaiTech. We have also trained the wOCC or WS model for 2 or 3 epochs in each training step, but obtained worse results, probably because training with more epochs would enhance the supervision of wrong pseudo labels in early training modules.

5 Conclusion

In this paper, we interleave the One-Classification and Weakly-Supervised models with adaptive thresholding to tackle unsupervised video anomaly detection

without any human annotations. We face the challenges of performance fluctuation and require of accurate threshold when designing the framework. To alleviate performance fluctuation, We propose the wOCC model which requires soft labels but not binary labels. To obtain a relative optimal threshold, we repeat our proposed interleaving training module, during which we propose adaptive thresholding to refine a rough threshold progressively. Extensive experiments demonstrate the effectiveness of our method. Remarkably, our method can be upgraded with the most recent development in OCC and WS VAD fields.

Acknowledgements

References

1. Acsintoae, A., Florescu, A., Georgescu, M., Mare, T., Sumedrea, P., Ionescu, R.T., Khan, F.S., Shah, M.: Ubnormal: New benchmark for supervised open-set video anomaly detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (June 2022)
2. Al-lahham, A., Tastan, N., Zaheer, M.Z., Nandakumar, K.: A coarse-to-fine pseudo-labeling (c2fpl) framework for unsupervised video anomaly detection. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. pp. 6793–6802 (2024)
3. Andrews, S., Tsochantaridis, I., Hofmann, T.: Support vector machines for multiple-instance learning. Advances in neural information processing systems **15** (2002)
4. Barbalau, A., Ionescu, R.T., Georgescu, M.I., Dueholm, J., Ramachandra, B., Nasrollahi, K., Khan, F.S., Moeslund, T.B., Shah, M.: Ssmtl++: Revisiting self-supervised multi-task learning for video anomaly detection. Computer Vision and Image Understanding **229**, 103656 (2023)
5. Basharat, A., Gritai, A., Shah, M.: Learning object motion patterns for anomaly detection and improved object detection. In: 2008 IEEE conference on computer vision and pattern recognition. pp. 1–8. IEEE (2008)
6. Burlina, P., Joshi, N., Wang, I., et al.: Where's wally now? deep generative and discriminative embeddings for novelty detection. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 11507–11516 (2019)
7. Dietterich, T.G., Lathrop, R.H., Lozano-Pérez, T.: Solving the multiple instance problem with axis-parallel rectangles. Artificial intelligence **89**(1-2), 31–71 (1997)
8. Feng, J.C., Hong, F.T., Zheng, W.S.: Mist: Multiple instance self-training framework for video anomaly detection. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 14009–14018 (2021)
9. Feng, X., Song, D., Chen, Y., Chen, Z., Ni, J., Chen, H.: Convolutional transformer based dual discriminator generative adversarial networks for video anomaly detection. In: Proceedings of the 29th ACM International Conference on Multimedia. pp. 5546–5554 (2021)
10. Georgescu, M.I., Barbalau, A., Ionescu, R.T., Khan, F.S., Popescu, M., Shah, M.: Anomaly detection in video via self-supervised and multi-task learning. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 12742–12752 (2021)

11. Georgescu, M.I., Ionescu, R.T., Khan, F.S., Popescu, M., Shah, M.: A background-agnostic framework with adversarial training for abnormal event detection in video. IEEE transactions on pattern analysis and machine intelligence **44**(9), 4505–4523 (2021)
12. Gong, D., Liu, L., Le, V., Saha, B., Mansour, M.R., Venkatesh, S., Hengel, A.v.d.: Memorizing normality to detect anomaly: Memory-augmented deep autoencoder for unsupervised anomaly detection. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 1705–1714 (2019)
13. Hirschorn, O., Avidan, S.: Normalizing flows for human pose anomaly detection. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 13545–13554 (2023)
14. Ionescu, R.T., Khan, F.S., Georgescu, M.I., Shao, L.: Object-centric auto-encoders and dummy anomalies for abnormal event detection in video. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 7842–7851 (2019)
15. Kay, W., Carreira, J., Simonyan, K., Zhang, B., Hillier, C., Vijayanarasimhan, S., Viola, F., Green, T., Back, T., Natsev, P., et al.: The kinetics human action video dataset. arXiv preprint arXiv:1705.06950 (2017)
16. Kingma, D.P., Dhariwal, P.: Glow: Generative flow with invertible 1x1 convolutions. Advances in neural information processing systems **31** (2018)
17. Li, S., Liu, F., Jiao, L.: Self-training multi-sequence learning with transformer for weakly supervised video anomaly detection. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 36, pp. 1395–1403 (2022)
18. Liu, W., Luo, W., Lian, D., Gao, S.: Future frame prediction for anomaly detection–a new baseline. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 6536–6545 (2018)
19. Liu, Z., Nie, Y., Long, C., Zhang, Q., Li, G.: A hybrid video anomaly detection framework via memory-augmented flow reconstruction and flow-guided frame prediction. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 13588–13597 (2021)
20. Lorre, G., Rabarisoa, J., Orcesi, A., Ainouz, S., Canu, S.: Temporal contrastive pretraining for video action recognition. In: Proceedings of the IEEE/CVF winter conference on applications of computer vision. pp. 662–670 (2020)
21. Lv, H., Yue, Z., Sun, Q., Luo, B., Cui, Z., Zhang, H.: Unbiased multiple instance learning for weakly supervised video anomaly detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 8022–8031 (June 2023)
22. Markovitz, A., Sharir, G., Friedman, I., Zelnik-Manor, L., Avidan, S.: Graph embedded pose clustering for anomaly detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 10539–10547 (2020)
23. Medioni, G., Cohen, I., Brémond, F., Hongeng, S., Nevatia, R.: Event detection and analysis from video streams. IEEE Transactions on pattern analysis and machine intelligence **23**(8), 873–889 (2001)
24. Morais, R., Le, V., Tran, T., Saha, B., Mansour, M., Venkatesh, S.: Learning regularity in skeleton trajectories for anomaly detection in videos. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 11996–12004 (2019)
25. Nguyen, T.N., Meunier, J.: Anomaly detection in video sequence with appearance-motion correspondence. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 1273–1283 (2019)

26. Pang, G., Yan, C., Shen, C., Hengel, A.v.d., Bai, X.: Self-trained deep ordinal regression for end-to-end video anomaly detection. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 12173–12182 (2020)
27. Park, H., Noh, J., Ham, B.: Learning memory-guided normality for anomaly detection. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 14372–14381 (2020)
28. Sapkota, H., Yu, Q.: Bayesian nonparametric submodular video partition for robust anomaly detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 3212–3221 (2022)
29. Shi, C., Sun, C., Wu, Y., Jia, Y.: Video anomaly detection via sequentially learning multiple pretext tasks. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 10330–10340 (2023)
30. Smeureanu, S., Ionescu, R.T., Popescu, M., Alexe, B.: Deep appearance features for abnormal behavior detection in video. In: Image Analysis and Processing-ICIAP 2017: 19th International Conference, Catania, Italy, September 11-15, 2017, Proceedings, Part II 19. pp. 779–789. Springer (2017)
31. Sohrab, F., Raitoharju, J., Gabbouj, M., Iosifidis, A.: Subspace support vector data description. In: 2018 24th International Conference on Pattern Recognition (ICPR). pp. 722–727. IEEE (2018)
32. Sultani, W., Chen, C., Shah, M.: Real-world anomaly detection in surveillance videos. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 6479–6488 (2018)
33. Sun, S., Gong, X.: Hierarchical semantic contrast for scene-aware video anomaly detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 22846–22856 (June 2023)
34. Tian, Y., Pang, G., Chen, Y., Singh, R., Verjans, J.W., Carneiro, G.: Weakly-supervised video anomaly detection with robust temporal feature magnitude learning. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 4975–4986 (2021)
35. Tran, D., Bourdev, L., Fergus, R., Torresani, L., Paluri, M.: Learning spatiotemporal features with 3d convolutional networks. In: Proceedings of the IEEE international conference on computer vision. pp. 4489–4497 (2015)
36. Tudor Ionescu, R., Smeureanu, S., Alexe, B., Popescu, M.: Unmasking the abnormal events in video. In: Proceedings of the IEEE international conference on computer vision. pp. 2895–2903 (2017)
37. Tur, A.O., Dall'Asen, N., Beyan, C., Ricci, E.: Exploring diffusion models for unsupervised video anomaly detection. In: 2023 IEEE International Conference on Image Processing (ICIP). pp. 2540–2544. IEEE (2023)
38. Wan, B., Fang, Y., Xia, X., Mei, J.: Weakly supervised video anomaly detection via center-guided discriminative learning. In: 2020 IEEE international conference on multimedia and expo (ICME). pp. 1–6. IEEE (2020)
39. Wang, G., Wang, Y., Qin, J., Zhang, D., Bao, X., Huang, D.: Video anomaly detection by solving decoupled spatio-temporal jigsaw puzzles. In: European Conference on Computer Vision. pp. 494–511. Springer (2022)
40. Wang, J., Song, Y., Leung, T., Rosenberg, C., Wang, J., Philbin, J., Chen, B., Wu, Y.: Learning fine-grained image similarity with deep ranking. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1386–1393 (2014)
41. Wang, J., Cherian, A.: Gods: Generalized one-class discriminative subspaces for anomaly detection. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 8201–8211 (2019)

42. Wang, Z., Zou, Y., Zhang, Z.: Cluster attention contrast for video anomaly detection. In: Proceedings of the 28th ACM international conference on multimedia. pp. 2463–2471 (2020)
43. Wu, J.C., Hsieh, H.Y., Chen, D.J., Fuh, C.S., Liu, T.L.: Self-supervised sparse representation for video anomaly detection. In: European Conference on Computer Vision. pp. 729–745. Springer (2022)
44. Wu, P., Liu, J., Shi, Y., Sun, Y., Shao, F., Wu, Z., Yang, Z.: Not only look, but also listen: Learning multimodal violence detection under weak supervision. In: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXX 16. pp. 322–339. Springer (2020)
45. Yan, C., Zhang, S., Liu, Y., Pang, G., Wang, W.: Feature prediction diffusion model for video anomaly detection. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 5527–5537 (October 2023)
46. Yang, Z., Liu, J., Wu, Z., Wu, P., Liu, X.: Video event restoration based on keyframes for video anomaly detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 14592–14601 (June 2023)
47. Zaheer, M.Z., Mahmood, A., Khan, M.H., Segu, M., Yu, F., Lee, S.I.: Generative cooperative learning for unsupervised video anomaly detection. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 14744–14754 (2022)
48. Zaheer, M.Z., Mahmood, A., Astrid, M., Lee, S.I.: Claws: Clustering assisted weakly supervised learning with normalcy suppression for anomalous event detection. In: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXII 16. pp. 358–376. Springer (2020)
49. Zhang, C., Li, G., Qi, Y., Wang, S., Qing, L., Huang, Q., Yang, M.H.: Exploiting completeness and uncertainty of pseudo labels for weakly supervised video anomaly detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 16271–16280 (June 2023)
50. Zhang, J., Qing, L., Miao, J.: Temporal convolutional network with complementary inner bag loss for weakly supervised anomaly detection. In: 2019 IEEE International Conference on Image Processing (ICIP). pp. 4030–4034. IEEE (2019)
51. Zhang, T., Lu, H., Li, S.Z.: Learning semantic scene models by object classification and trajectory clustering. In: 2009 IEEE conference on computer vision and pattern recognition. pp. 1940–1947. IEEE (2009)
52. Zhao, Y., Deng, B., Shen, C., Liu, Y., Lu, H., Hua, X.S.: Spatio-temporal autoencoder for video anomaly detection. In: Proceedings of the 25th ACM international conference on Multimedia. pp. 1933–1941 (2017)
53. Zhong, J.X., Li, N., Kong, W., Liu, S., Li, T.H., Li, G.: Graph convolutional label noise cleaner: Train a plug-and-play action classifier for anomaly detection. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 1237–1246 (2019)