Graph Convolutional Label Noise Cleaner: Train a Plug-and-play Action Classifier for Anomaly Detection

Jia-Xing Zhong[1,2] Nannan Li[3,1,2] Weijie Kong[1,2] Shan Liu[4] Thomas H. Li[1] Ge Li ✉[1,2]
[1]School of Electronic and Computer Engineering, Peking University [2]Peng Cheng Laboratory
[3]Institute of Intelligent Video Audio Technology, Longgang Shenzhen [4]Tencent America
jxzhong@pku.edu.cn lnnsiat@gmail.com weijie.kong@pku.edu.cn
shanl@tencent.com tli@aiit.org.cn geli@ece.pku.edu.cn

Abstract

Video anomaly detection under weak labels is formulated as a typical multiple-instance learning problem in previous works. In this paper, we provide a new perspective, i.e.*, a supervised learning task under noisy labels. In such a viewpoint, as long as cleaning away label noise, we can directly apply fully supervised action classifiers to weakly supervised anomaly detection, and take maximum advantage of these well-developed classifiers. For this purpose, we devise a graph convolutional network to correct noisy labels. Based upon feature similarity and temporal consistency, our network propagates supervisory signals from high-confidence snippets to low-confidence ones. In this manner, the network is capable of providing cleaned supervision for action classifiers. During the test phase, we only need to obtain snippet-wise predictions from the action classifier without any extra post-processing. Extensive experiments on 3 datasets at different scales with 2 types of action classifiers demonstrate the efficacy of our method. Remarkably, we obtain the frame-level AUC score of 82.12% on UCF-Crime.*

1. Introduction

Anomaly detection in videos has been long studied for its ubiquitous applications in real-world scenarios, *e.g.* intelligent surveillance, violence alerting, evidence investigation, *etc*. Since anomalous events are rarely seen in common environments, anomalies are often defined as behavioral or appearance patterns different from usual patterns in previous work [6, 1, 13]. Based on this definition, a popular paradigm for anomaly detection is one-class classification [66, 11] (a.k.a. unary classification), *i.e.*, to encode the usual pattern with only normal training samples. Then the distinctive encoded patterns are detected as anomalies. However, it is impossible to collect all normal behaviors in a dataset. Therefore some normal events might deviate from the encoded patterns, and could cause false alarms. In recent years, there has been some research [20, 22, 58] on an emerging *binary-classification paradigm*: the training data contain both anomalous and normal videos.

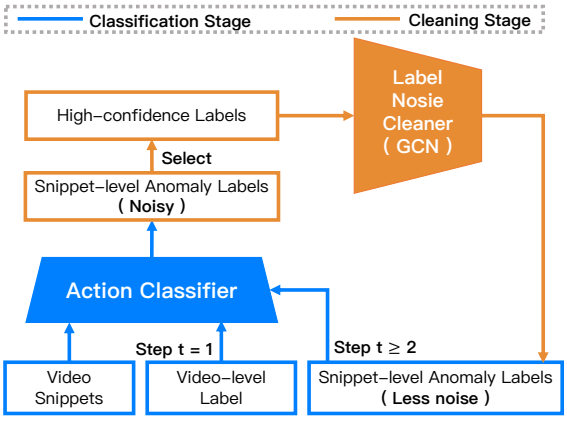


Figure 1: *The concept of alternate optimization mechanism.* Noisy labels predicted by the action classifier are utilized to train the label noise cleaner and then they are refined. The cleaned labels are reassigned to optimize the action classifier. The two training processes are executed alternatively.

Following the binary-classification paradigm, we attempt to address the *weakly supervised anomaly detection* problem, on which only video-level anomaly labels are available in the training data. In this problem, there are neither trimmed anomalous segments nor temporal annotations for the consideration of the human-labor cost.

The weakly supervised anomaly detection problem is viewed as a multiple-instance learning (MIL) task in prior works [20, 22, 58]. They consider a video (or a set of snippets) as a bag, which consists of the snippets (or frames) deemed as instances, and learn instance-level anomaly labels via bag-level annotations. In this paper, we address

the problem from a new perspective, formulating it as a supervised learning task under noise labels. The *noise labels* refer to wrong annotations of normal snippets within anomalous videos, since a video labeled as "anomaly" may contain quite a few normal clips. In such a viewpoint, we can directly train fully supervised action classifiers once the noisy labels are cleaned.

There are noticeable advantages of our noise-labeled perspective *in both the training and the test phase*. Instead of simply extracting offline features for MIL models, our action classifier participates in the whole learning process. During the training process, the only difference between action classifiers and fully supervised updating is the input labels. As a result, we preserve all strengths of theses action classifiers, such as well-designed structures, transferable pre-trained weights, ready-to-use source codes, *etc*. As for testing, the trained classifier can directly make predictions without any post-processing. It is extremely convenient and highly efficient because the feature extraction and the abnormality decision are seamlessly integrated into a single model.

Intuitively, a well-trained classifier yields the predictions with less noise, and the cleaned labels in turn help to train a better classifier. To this end, we design an alternate training procedure as Figure 1 illustrates. It consists of two alternate stages, *i.e*., cleaning and classification. In the cleaning stage, we train a cleaner to correct the noisy predictions obtained from the classifier, and the cleaner provides refined labels with less noise. In the classification stage, the action classifier is retrained with the cleaned labels and generates more reliable predictions. Such a cyclic operation is executed several times until convergence. The main idea of our cleaner is to eliminate noise of low-confidence predictions via high-confidence ones. We devise a graph convolutional network (GCN) to establish relationships between high-confidence snippets and low-confidence ones. In the graph, snippets are abstracted into vertexes and the anomaly information is propagated through edges. During testing, we no longer require the cleaner and directly obtain snippet-wise anomaly results from the trained classifier. For verification of the general applicability of our model, we carry out extensive experiments with two types of mainstream action classifiers: a 3D-conv network C3D [59] and a two-stream structure TSN [62]. In addition, we evaluate the proposed approach on 3 different-scale datasets, *i.e*., UCF-Crime [58], ShanghaiTech [43] and UCSD-Peds [35]. The experimental results demonstrate that our model advances the state-of-the-art performance of weakly supervised anomaly detection.

In a nutshell, the contribution of this paper is three-fold:

- We formulate the problem of anomaly detection with weak labels as a supervised learning task under noise annotations, and put forward an alternate training framework to optimize the action classifier.
- We propose a GCN to clean noise labels. To the best of our knowledge, it is the first work to apply a GCN to correct label noise in the area of video analytics.
- We conduct experiments on 3 different-scale anomaly detection datasets with two types of action classifiers, in which the state-of-the-art performance validates the effectiveness of our approach. The source code is available at https://github.com/jx-zhong-for-academic-purpose/GCN-Anomaly-Detection.

2. Related Work

Anomaly detection. As one of the most challenging problem, anomaly detection in videos has been extensively studied for many years [30, 67, 65, 19, 34, 3, 47, 35] . Most research addresses the problem under the assumption that anomalies are rare or unseen, and behaviors deviating from normal patterns are supposed to be anomalous. They attempt to encode regular patterns via a variety of statistic models, *e.g*. the social force model [45], the mixture of dynamic models on texture [35], Hidden Markov Models on video volumes [21, 30], the Markov Random Field upon spatial-temporal domain [28], Gaussian process modeling [49, 11], and identify anomalies as outliers. Sparse reconstruction [41, 31, 13, 67] is also another popular approach for usual pattern modeling. They utilize sparse representation to construct a dictionary for normal behavior, and detect anomalies as the ones with high reconstruction error. Recently, with the great success of deep learning, a few researchers design deep neural networks on abstraction feature learning [19, 12, 42] or video prediction learning [40] for anomaly detection. As opposed to the works that built their detection models on normal behavior only, there is research [2, 20, 58] employing both usual and unusual data for model building. Among them, MIL is used for motion pattern modeling under weakly supervised setting [20, 58]. Sultani *et al*. [58] propose an MIL-based classifier to detect anomalies, where a deep anomaly ranking model predicts anomaly scores. Unlike them, we formulate the anomaly detection problem with weak labels as a supervised learning under noise labels, and devise an alternate training procedure to progressively promote the discrimination of action classifiers.

Action analysis. Action classification is a long standing problem in the field of computer vision, and a large body of research works [61, 59, 62, 10, 26, 63] have been presented. A majority of modern approaches have introduced deep architecture models [10, 59, 57, 62], including the most prevailing two-stream networks [57], C3D [59] and their variants [62, 15, 53, 10]. Up to now, deep learning based methods have achieved state-of-the-art performance. Besides action classification, some researchers recently have

focused on temporal action localization [68, 38, 69, 56, 16]. The performance metrics of temporal action detection and anomaly detection are quite different: action detection aims to find a temporal interval overlapped with the ground truth as much as possible, whereas anomaly detection aims for a robust frame-level performance under various discrimination thresholds. In this paper, we attempt to leverage the powerful action classifiers to detect anomalies in a simple and feasible way.

Learning under noisy labels. The research works [33, 48, 51, 17] addressing the noise label problem can be generally divided into two categories: noise reduction and loss correction. In the case of noise reduction, they aim to correct noisy labels via formulating the noise model explicitly or implicitly, such as Conditional Random Fields (CRF) [60], knowledge graphs [37]. Approaches in the latter group are developed for directly learning with label noise, utilizing correction methods for loss adjustment. Azadi *et al*. [4] actively select training features via imposing a regularization term on loss function. Different from theses general approaches, our GCN is intended for videos and take advantages of the video-based characteristics.

Graph convolutional neural network. In recent years, a surge of graph convolutional networks [50, 29, 52, 36, 18] have been proposed to tackle graph-structured data. An important stream of these works is utilizing spectral graph theory [8, 14], which decomposes the graph signal on the spectral domain and defines a series of parameterized filters for convolution. A number of researchers propose improvements of spectral convolutions, leading to advanced performances on tasks such as node classification and recommendation system. The goal of our label noise cleaner is classifying nodes (video snippets) in a graph (the whole video) under the supervision of high-confidence annotations.

3. Problem Statement

Given a video $V = \{v_i\}_{i=1}^{N}$ with N snippets, the observable label $Y \in \{1, 0\}$ indicates whether this video contains anomalous clips or not. Note that no temporal annotation is provided in training data. The goal of anomaly detection is to pinpoint the temporal position of abnormalities once they occurs in test videos.

Sabato and Tishby [54] provide a theoretical analysis in which MIL tasks can be viewed as learning under one-sided label noise. In some prior works [20, 22, 58], anomaly detection under the weak supervisory signal is described as a typical MIL problem. Therefore, we naturally cast anomaly detection from MIL formulation to noisy label setting.

MIL formulation. In this formulation, each clip v_i is considered as an *instance*, of which the anomaly label y_i is unavailable. These clips compose the *positive/negative bag* according to the given video-level anomaly label Y: a *positive bag* $(Y = 1)$ includes at least one anomalous clip, while a *negative bag* $(Y = 0)$ is entirely comprised of normal snippets. Consequently, anomaly detection is modeled as key instance detection [39] under MIL, in search of positive instances v_i with $y_i = 1$. This MIL setting allows learning instance-level labels under bag-level supervision, and a set of approaches [20, 22, 58] is derived from this.

Noisy-labeled learning formulation. It is evident that the label $Y = 0$ is noiseless, since it means all snippets v_i in the video V are normal:

$$Y = 0 \Rightarrow y_i = 0,\ \forall v_i \in V\,. \tag{1}$$

However, $Y = 1$ is noisy because in this case the video V is partially made up of anomalous clips:

$$Y = 1 \not\Rightarrow y_i = 1,\ \forall v_i \in V\,. \tag{2}$$

This is referred to as *one-sided label noise* [7, 9, 55], for the noise only appears along with $Y = 1$. As long as appropriately handling the label noise w.r.t. $Y = 1$, we are able to readily apply a variety of well-developed action classifiers to anomaly detection.

4. Graph Convolutional Label Noise Cleaner

Similar to many noisy-labeled learning approaches, our method adopts an EM-like optimization mechanism: alternately training the action classifier and the noise cleaner. At each training step of the noise cleaner, we have obtained rough snippet-wise anomaly probabilities from the action classifier, and the target of our noise cleaner is to *correct low-confidence anomaly scores via high-confidence ones.*

Unlike other general noise-labeled learning algorithms, our cleaner is specifically designed for videos. To the best of our knowledge, this is the first work to deploy a GCN in noise-labeled videos. In the graph convolutional network, we leverage two characteristics of a video to correct the label noise, *i.e*., *feature similarity* and *temporal consistency*. Intuitively, *feature similarity* means the anomaly snippets share some similar characteristics, while *temporal consistency* means anomaly snippets probably appear in temporal proximity of each other.

4.1. Feature Similarity Graph Module

As Figure 2 depicts, features from the action classifier are first compressed with two fully connected layers to mitigate the curse of dimensionality [5]. We model the feature similarly with an attributed graph [52] $F = (V, E, \mathbf{X})$, where V is the vertex set, E is the edge set, and $\mathbf{X}$ is the attribute of vertexes. In particular, V is a video as defined in Section 3, E describes the feature similarity amongst snippets, and $\mathbf{X} \in \mathbb{R}^{\mathbf{N}\times\mathbf{d}}$ represents the d-dimensional feature of these N snippets. The adjacency matrix $\mathbf{A^F} \in \mathbb{R}^{N\times N}$ of F is defined as:

$$\mathbf{A^F}_{(i,j)} = \exp(\mathbf{X}_i \cdot \mathbf{X}_j - max(\mathbf{X}_i \cdot \mathbf{X}))\,, \tag{3}$$

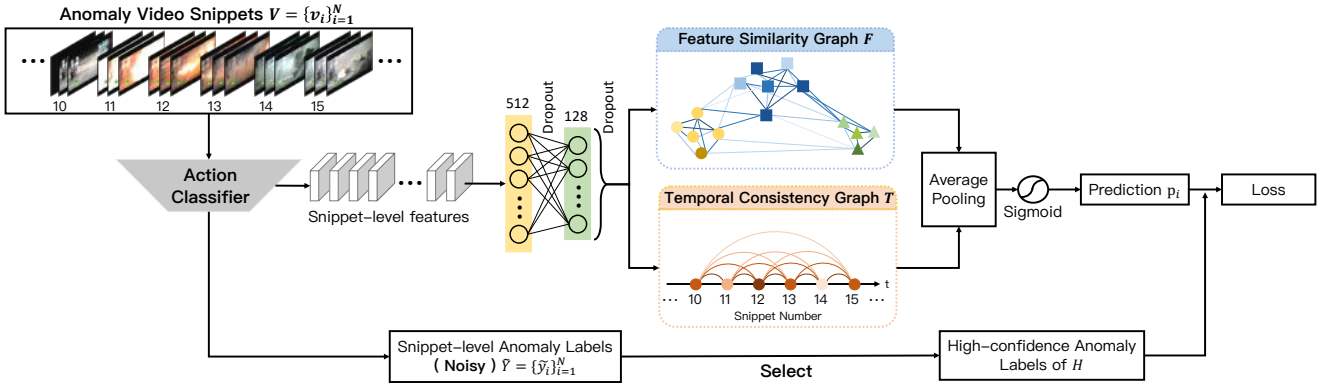


Figure 2: *Overview of the training process of label noise cleaner.* The action classifier extracts spatio-temporal features from anomalous video snippets and outputs noisy snippet-level labels. Snippet-level features from the classifier are compressed and fed into two graph modules to model the feature similarity and temporal consistency of snippets. In the two graph-based modules, A darker node represents higher anomaly confidence of the snippet. The output of these two models are fused and utilized to predict the snippet-level labels with less noise. The loss is updated to correct the predictive noise via high-confidence snippets.

where the element $\mathbf{A^F}_{(i,j)}$ measures the feature similarly between the i^{th} and j^{th} snippets. Since an adjacency matrix should be non-negative, we bound the similarity to the range $(0, 1]$ with a normalized exponential function. Based on the graph F, snippets with similar features are closely connected, and the label assignments are propagated differently in accordance with different adjacency values.

The nearby vertexes are driven to have the same anomaly label via graph-Laplacian operations. Following Kipf and Welling [29], we approximate the graph-Laplacian with a renormalization trick:

$$\widehat{\mathbf{A}}^{\mathbf{F}} = \widetilde{\mathbf{D}}^{\mathbf{F}-\frac{1}{2}} \widetilde{\mathbf{A}}^{\mathbf{F}} \widetilde{\mathbf{D}}^{\mathbf{F}-\frac{1}{2}}, \tag{4}$$

where the self-loop adjacency matrix $\widetilde{\mathbf{A}}^{\mathbf{F}} = \mathbf{A^F} + \mathbf{I_n}$, and $\mathbf{I_n} \in \mathbb{R}^{N\times N}$ is the identity matrix; $\widetilde{\mathbf{D}}^{\mathbf{F}}_{(i,i)} = \sum_j \widetilde{\mathbf{A}}^{\mathbf{F}}_{(i,j)}$ is the corresponding degree matrix. Finally, the output of a feature similarity graph module layer is computed as:

$$\mathbf{H^F} = \sigma(\widehat{\mathbf{A}}^{\mathbf{F}}\mathbf{X}\mathbf{W}), \tag{5}$$

where $\mathbf{W}$ is a trainable parametric matrix, and σ is an activation function. Since the whole computational procedure is differentiable, our feature similarity graph module can be trained in an *end-to-end* fashion. Therefore, neural networks are capable of seamlessly incorporating the single or multiple stacked modules. Although the aforementioned procedure contains some element-wise calculations, we provide a high-efficient vectorized implementation in **Appendix**.

Recently, Wang and Gupta [63] also have established similarity graphs to analyze a video. Nevertheless, both the goal and the method are quite different from ours: they aim to capture long-term dependencies with the similarity relations of correlated objects/regions, whereas we attempt to propagate supervisory signals with the similarity levels of entire snippets/frames.

4.2. Temporal Consistency Graph Module

As pointed out in [24, 46, 64], temporal consistency is advantageous to many video-based tasks. The temporal consistency graph T is directly built upon the temporal structure of a video. Its adjacency matrix $\mathbf{A^T} \in \mathbb{R}^{N\times N}$ is only dependent on temporal positions of the i^{th} and j^{th} snippets:

$$\mathbf{A^T}_{(i,j)} = k(i,j), \tag{6}$$

where k is a non-negative kernel function. Consider that the kernel is supposed to distinguish various temporal distances and closely connect the snippets in vicinity. In practice, we use an exponential kernel (a.k.a. Laplacian kernel) neatly bounded in $(0, 1]$:

$$k(i,j) = \exp(-||i-j||). \tag{7}$$

Likewise, we obtain the renormalized adjacency matrix $\widehat{\mathbf{A}}^{\mathbf{T}}$ as Equation 4 for the graph-Laplacian approximation, and the forward result of this module is computed as:

$$\mathbf{H^T} = \sigma(\widehat{\mathbf{A}}^{\mathbf{T}}\mathbf{X}\mathbf{W}), \tag{8}$$

where $\mathbf{W}$ is a trainable parametric matrix, σ is an activation function, and $\mathbf{X}$ is the input feature matrix. The stacked temporal consistency graph layers also can be conveniently included into neural networks.

4.3. Loss Function

Finally, the outputs of the above two modules are fused with an average pooling layer, and activated by a Sigmoid function to make the probabilistic prediction p_i of each vertex in the graph, corresponding to the anomaly probability of our noise cleaner w.r.t. the i^{th} snippet. The loss function $\mathcal{L}$ is based upon two types of supervision:

$$\mathcal{L} = \mathcal{L}_D + \mathcal{L}_I \,, \tag{9}$$

where $\mathcal{L}_D$ and $\mathcal{L}_I$ are computed under the *direct* and the *indirect* supervision respectively. Given the rough snippet-wise anomaly probabilities $\widetilde{Y} = \{\widetilde{y}_i\}_{i=1}^N$ from the action classifier. The loss term under *direct supervision* is defined as a cross-entropy error over the high-confidence snippets:

$$\mathcal{L}_D = -\frac{1}{|H|} \sum_{i \in H} [\widetilde{y}_i \ln p_i + (1 - \widetilde{y}_i) \ln (1 - p_i)] \,, \tag{10}$$

where H is the set of high-confidence snippets. We over-sample each video frame with the "10-crop" augment,[1] and calculate mean anomaly probabilities $\widetilde{y}_i$ as well as predictive variances of the action classifier. As pointed out by Kendall and Gal [27], variance measures the uncertainty of predictions. In other words, *the smaller variance indicates the higher confidence.* This criterion of confidence is conceptually simple yet practically effective.

The *indirectly supervised term* is a temporal-ensembling strategy [32] to further harness a small number of labeled data, because high-confidence predictions are only from a portion of the entire video. Its main idea is to smooth the network predictions of all snippets at different training steps:

$$\mathcal{L}_I = \frac{1}{N} \sum_{i=1}^{N} |p_i - \overline{p}_i| \,, \tag{11}$$

where $\overline{p}_i$ is the discount-weighted average predictions of our noise cleaner over various training epochs. There is a major difference between the original "cool start" initialization and our implementation as explained in **Appendix**, since we have already obtained a set of rough predictions from the action classifier.

4.4. Alternate Optimization

The training process of our noise cleaner is merely one part of the alternate optimization. The other part, *i.e.*, the training process of our classifier, is exactly the same as common fully supervised updating, *except that the labels are snippet-wise predictions from our trained cleaner*. After repeating such an alternate optimization several times, final anomaly detection results are directly predicted by the last trained classifier. Obviously, almost no change in the action classifier is required during the training or the test phase. As a result, we can conveniently train the fully supervised action classifier under weak labels, and directly deploy it for anomaly detection without all the bells and whistles.

[1] "10-crop" means cropping images into the center, four corners, and their mirrored counterparts.

5. Experiments

5.1. Datasets and Evaluation Metric

We conduct the experiments upon three datasets of various scales, *i.e.*, UCF-Crime [58], ShanghaiTech [43] and UCSD-Peds [35].

UCF-Crime is a large-scale dataset of real-world surveillance videos. It has 13 types of anomalies with 1,900 long untrimmed videos, which consist of 1,610 training videos and 290 test videos.

ShanghaiTech is a medium-scale dataset of 437 videos, including 130 abnormal events on 13 scenes. In the standard protocol [43], all training videos are normal, and this setting is inappropriate for the binary-classification task. Hence, we reorganize the dataset by randomly selecting anomaly testing videos into training data and vice versa. Meanwhile, both training videos and testing ones cover all of the 13 scenes. This new split of the dataset will be available for follow-up comparisons. More details are given in **Appendix**.

UCSD-Peds is a small-scale dataset made up of two subsets: Peds1 has 70 videos, and Peds2 has 28 videos. Since the former is more frequently used for pixel-wise anomaly detection [66], we only conduct experiments on the latter as in [43]. Similarly, the default training set does not contain anomaly videos. Following He *et al*. [20], 6 anomaly videos and 4 normal ones on UCSD-Peds2 are randomly included into training data, and the remaining videos constitute the test set. We also repeat this process 10 times and report the average performance.

Evaluation Metric. Following previous works [43, 20, 58], we plot the frame-level receiver operating characteristics (ROC) curve and compute an area under the curve (AUC) as the evaluation metric. In the task of temporal anomaly detection, a larger frame-level AUC implies the higher diagnostic ability, as well as the robuster performance at various discrimination thresholds.

5.2. Implementation Details

Action classifiers. For verification of the general applicability of our model, we utilize two mainstream structures of action classifiers in the experiments. **C3D** [59] is a *3D-convolutional* network. The model is pre-trained on the Sports-1M [26] dataset. In the training process, we input features from its *fc7* layer into our label noise cleaner. **Temporal Segment Network (TSN)** [62] is a *two-stream* architecture. We choose BN-Inception [23] pre-trained on

Table 1: *Ablation Studies on UCF-Crime.*

Training Stage	Indirect Supervision	Temporal Consistency		Feature Similarity		AUC (%)
		Conv.	Graph	Conv.	Graph	
Step-2	√	√	√	√	√	74.60
Step-2		√	√	√	√	73.79
Step-2		√				67.57
Step-2		√	√			72.93
Step-2				√		67.23
Step-2				√	√	72.44
Step-1	–	–	–	–	–	70.87

Table 2: *Quantitative comparison on UCF-Crime.* † and ‡ indicate the loss without and with constraints respectively.

Method	AUC (%)	False Alarm (%)
SVM Baseline	50.0	–
Hasan *et al.* [19]	50.6	27.2
Lu *et al.* [41]	65.51	3.1
Sultani *et al.* † [58]	74.44	–
Sultani *et al.* ‡ [58]	75.41	1.9
Ours		
C3D	81.08	2.8
TSN^{RGB}	82.12	0.1
$\text{TSN}^{OpticalFlow}$	78.08	1.1

Kinetics-400 [10] as the backbone, and extract features from its *global_pool* layer to train our noise cleaner. The action classifiers are both implemented upon the Caffe [25] platform with the same settings of video sampling and data augment as [62]. In all the experiments, we keep the default settings if not specified particularly.

Label noise cleaner. After we add the author list and the acknowledgement section into our camera-ready version, this part has to be moved to **Appendix** because of limited space. Please refer to our Github page and **Appendix**.

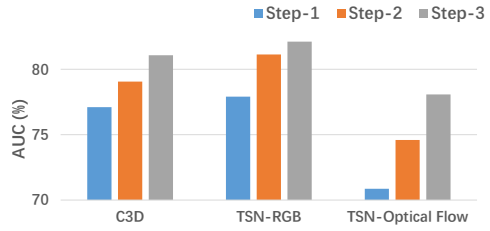


Figure 3: *Step-wise performance on UCF-Crime.*

5.3. Experiments on UCF-Crime

Under the video-level supervision, we train C3D with 18,000 iterations. As for TSN, the initial iteration number of both streams is 20,000. At each re-training step, we stop the updating procedure at 4,000 iterations.

Step-wise results. As Figure 3 depicts, we report the AUC performance at each step to evaluate the efficacy of our alternate training mechanism. Even if only given video-level labels, C3D and the RGB branch of TSN can achieve a descent performance at the Step-1. It is a wise choice for us to involve action classifiers in the training process. However, the optical flow stream of TSN is far from satisfaction, which reflects the necessity of our noise cleaner. At the following steps, the proposed approach significantly improves the detection performance of all the action classifiers. Faced with the most noise in initial predictions, the AUC performance of our optical flow branch is still boosted from 70.87% to 78.08% with a relative gain of 10.2%.

Indirect supervision. We conduct ablation studies upon the optical flow modality of TSN. First, we exclude the indirectly supervised term from the loss to verify its effectiveness. As on the 2^{nd} row of Table 1, the performance slightly declines from 74.60% to 73.79%, but the gain on the result of Step-1 remains considerable. In the following ablations, we remove the indirect supervised term to eliminate interference.

Temporal consistency. We would like to explore two questions: *Is temporal information helpful? Can our graph convolution utilize this information?* By excluding the other interference factors, there is only the temporal consistency module. To remove the graph of temporal information, we fill the $\mathbf{A^T}$ in Equation 6 with 0.5 (the mid-value of its bounds) and reproduce the alternate training procedure. As shown on the 3^{rd} row of Table 1, the performance without temporal graph is worse than that of Step-1, in which case the GCN only memorizes the pattern of high-confidence predictions but ignores other snippets. As for the ablation on graph convolution, we observe that the independent temporal consistency module boosts the AUC to 72.93% as on the 4^{th} row of Table 1, which demonstrates that our graph convolution really capitalizes on the temporal information.

Feature similarity. Likewise, we only reserve the feature similarity module to investigate the efficacy of similarity graphs and our convolutional operation. We first damage the feature similarity graph by setting all elements of the adjacency matrix as the mid-value. As on the 5^{th} row of Table 1, the AUC value falls to 67.23% without the graph. After recovering the original feature similarity graph, the single feature similarity module can increase the AUC value from 70.87% to 72.44% as shown on the 6^{th} row of Table 1. This illustrates that both similarity graphs and the convolution are beneficial to clean the noisy labels.

Quantitative comparison. We compare our methods with state-of-the-art models upon 3 indicators, *i.e.*, ROC

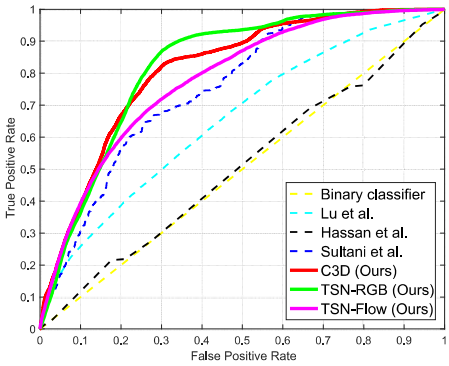


Figure 4: *ROC curves on UCF-Crime*.

Table 3: *Step-wise AUC (%) on ShanghaiTech*.

Action Classifier	C3D	TSN$^{\text{RGB}}$	TSN$^{\text{Optical Flow}}$
Step-1	73.79	80.83	78.23
Step-2	76.16	82.17	84.19
Step-3	76.44	84.44	84.13

curves, AUC and false alarm rates. As Figure 4 shows, our curves of all the action classifiers almost completely enclose the others, which means they are consistently superior to their competitors at various thresholds. The smoothness of the three curves shows the high stability of our proposed approach. As shown in Table 2, we boost the AUC value up to 82.12% at most. As for false alarm rates at 0.5 detection score, the C3D is slightly inferior to Sultani *et al*., whereas the other two classifiers are fairly satisfactory as shown in Table 2. Notably, the RGB branch of TSN reduces the false alarm rate to 0.1%, nearly 1/20 of the best-so-far result.

Qualitative analysis on the test set. To observe the influence of our model, we visualize the before-and-after change in predictions of action classifiers. As presented in Figure 5, our denoising process substantially alleviates the predictive noise of action classifiers within both normal and anomaly snippets. Intriguingly, the classifier fails to detect the anomaly event in the "Arrest007" video from beginning to end as Figure 5c depicts. After watching all videos of the "Arrest" class, we finally discover the possible cause: the similar scene in this testing video does not exist in training data. In this video, a man is arrested at the laundromat for vandalism of washing machines as shown in Figure 5d, while "Arrest" events occur on the highway or at the checkout counter in training data. It implies that to detect anomalous events in generic scenes is still a big challenge for the limited generalization ability of current models.

5.4. Experiments on ShanghaiTech

Step-wise results. As illustrated in Table 3, the performance is improved after the alternate training w.r.t. all the action classifiers. The results of optical flow branch of TSN at Step-3 reflects that excessive iterations may deteriorates on the detection performance. Nevertheless, our method performs robustly as the AUC value only drops slightly.

Table 4: *Quantitative comparison on UCSD-Peds2*. Following the reviewer comment, we make more comparisons as shown in **Appendix**.

Method	AUC (%)
Adam [1]	63.0
MDT [44]	85.0
SRC [13]	86.1
AMDN [66]	90.8
AL [20]	90.1
Ours	
TSN$^{Gray-scale}$	93.2 ± 2.3
TSN^{OpticalFlow}	92.8 ± 1.6

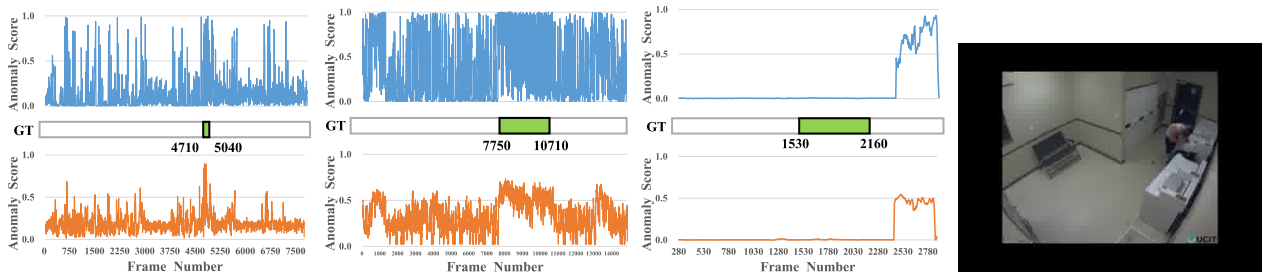
Qualitative Analysis. Different from UCF-Crime, the training data in the new split of ShanghaiTech have temporal ground truths. Based on this, the working principle of our GCN can be intuitively understood. The anomaly event in Figure 6 is that a student jumps over the rail as shown in Figure 7. The temporal consistency module (at the upper right) is inclined to smooth the original high-confidence predictions (orange points at the upper left). Therefore, it correctly annotates the $150^{th} - 200^{th}$ frames with dense high-confidence predictions, but neglects the remaining ground truth for insufficient high-confidence inputs. The feature similarity module (at the lower right) tends to propagate information through a similar degree. It labels a long interval of snippets including the student's previous run-up and subsequent slow-down actions, possibly because they have the similar representation of "a fast movement in the same direction" on the optical flow. The entire GCN (at the lower left) combining these two modules can make more precise labels.

5.5. Experiments on UCSD-Peds

In UCSD-Peds, some of the ground truths are only 4 frames, but the predictive unit of C3D reaches a length of 16 frames. Thus we conduct the experiments with TSN. To match the input dimension with the RGB branch, the original gray-scale frames are duplicated into the 3 primary-color channels.

Step-wise results. After repeating experiments 10 times, we obtain the box plots in Figure 8. The average results at the first step are good enough, so we start with feeding top 90% high-confidence predictions into the GCN. We observe that the proposed method not only increases the detection performance, but also stabilize the predictions of the 10-time repeated experiments.

Quantitative comparison. We report the "mean value



(a) Burglary005 (b) Burglary079 (c) Arrest007 (d) Arrest007 (Partial Ground Truth)

Figure 5: *Visualization of testing results on UCF-Crime.* The blue curves are predictions of the action classifier trained under video-level labels, and the orange curves are the results under cleaned supervision. The "GT" bars in green are ground truths. Best viewed in Adobe Reader where (d) should play as a video.

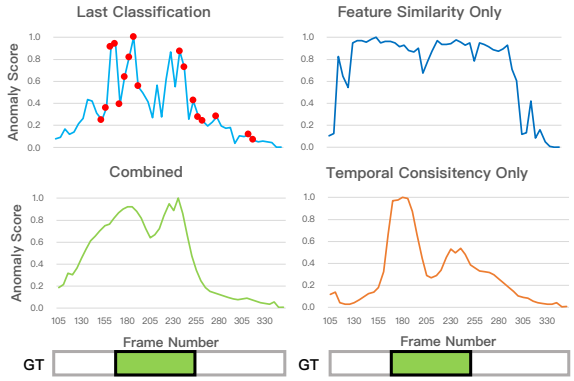
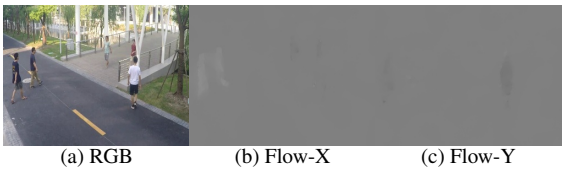


Figure 6: *Visualization of GCN outputs on ShanghaiTech w.r.t. the video "05_0021".* The rough prediction at the upper left is from the optic flow branch, while the other three are snippet-wise labels cleaned by the GCN modules.



(a) RGB (b) Flow-X (c) Flow-Y

Figure 7: *Partial video of "05_0021" on ShanghaiTech.* Best viewed in Adobe Reader where (a)-(c) should play as videos.

± standard deviation" of the AUC, and make comparisons with other methods under the same splitting protocol as in [20]. Our approach outperform others with both the input modalities as shown in Table 4.

6. Conclusion

In this paper, we address weakly supervised anomaly detection from a new perspective, by casting it as a supervised learning task under noise labels. In contrast to MIL formulation in previous works, such a perspective possesses distinct merits in two aspects: a) it directly inherits all the strengths of well-developed action classifiers; b) anomaly detection is accomplished by an integral end-to-end model with great convenience. Furthermore, we utilize a GCN to clean labels for training an action classifier. During the alternate optimization process, the GCN reduces noise via propagating anomaly information from high-confidence predictions to low-confidence ones. We validate the proposed detection model on 3 different-scale datasets with 2 types of action classification networks, where the superior performance proves its effectiveness and versatility.

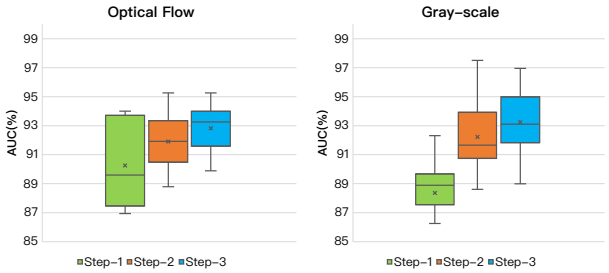


Figure 8: *Box-whisker plots of step-wise performance on UCSD-Peds2.*

Acknowledgement. This work was supported in part by the Project of National Engineering Laboratory-Shenzhen Division for Video Technology, in part by the National Natural Science Foundation of China and Guangdong Province Scientific Research on Big Data (No. U1611461), in part by Shenzhen Municipal Science and Technology Program (Grant JCYJ20170818141146428), and in part by National Natural Science Foundation of China (No. 61602014). We are grateful to the three anonymous reviewers for their valuable comments and suggestions. In addition, we would like to thank Jerry for English language editing.

References

[1] A. Adam, E. Rivlin, I. Shimshoni, and D. Reinitz. Robust real-time unusual event detection using multiple fixed location monitors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30:555–560, 2008.

[2] K. Adhiya, S. Kolhe, and S. Patil. Tracking and identification of suspicious and abnormal behaviors using supervised machine learning technique. In *Proceedings of the International Conference on Advances in Computing, Communication and Control*, pages 96–99, 2009.

[3] B. Anti and B. Ommer. Video parsing for abnormality detection. In *CVPR*, pages 2415–2422, Nov. 2011.

[4] S. Azadi, J. Feng, S. Jegelka, and T. Darrell. Auxiliary image regularization for deep cnns with noisy labels. In *ICLR*, 2016.

[5] Richard Bellman. *Dynamic Programming*. Princeton University Press, Princeton, NJ, USA, 1 edition, 1957.

[6] Y. Benezeth, P. Jodoin, V. Saligrama, and C. Rosenberger. Abnormal events detection based on spatio-temporal co-occurences. In *CVPR*, pages 2548–2465, 2009.

[7] Avrim Blum and Adam Kalai. A note on learning from multiple-instance examples. *Machine Learning*, 30(1):23–29, Jan 1998.

[8] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun. Spectral networks and locally connected networks on graphs. In *ICLR*, 2014.

[9] Marc-Andr Carbonneau, Veronika Cheplygina, Eric Granger, and Ghyslain Gagnon. Multiple instance learning: A survey of problem characteristics and applications. *Pattern Recognition*, 77:329 – 353, 2018.

[10] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *CVPR*, pages 4724–4733. IEEE, 2017.

[11] Kai-Wen Cheng, Yie-Tarng Chen, and Wen-Hsien Fang. Video anomaly detection and localization using hierarchical feature representation and gaussian process regression. In *CVPR*, June 2015.

[12] Y. S. Chong and Y. H. Tay. Abnormal event detection in videos using spatiotemporal autoencoder. In *International Symposium on Neural Networks*, pages 189–196, June 2017.

[13] Y. Cong, J. Yuan, and J. Liu. Sparse reconstruction cost for abnormal event detection. In *CVPR*, pages 3449–3456, 2011.

[14] M. Defferrard, X. Bresson, and P. Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *NIPS*, 2016.

[15] C. Feichtenhofer, A. Pinz, and A. Zisserman. Convolutional two-stream network fusion for video action recognition. In *CVPR*, pages 1933–1941, 2016.

[16] Jiyang Gao, Zhenheng Yang, and Ram Nevatia. Cascaded boundary regression for temporal action detection. In *BMVC*, 2017.

[17] J. Goldberger and E. Ben-Reuven. Training deep neuralnetworks using a noise adaptation layer. In *ICLR*, 2017.

[18] A. Grover and J. Leskovec. Node2vec: Scalable feature learning for networks. In *KDD*, 2016.

[19] M. Hasan, J. Choi, J. Neumann, A. K. Roy-Chowdhury, and L. S. Davis. Learning temporal regularity in video sequences. In *CVPR*, pages 733–742, June 2016.

[20] Chengkun He, Jie Shao, and Jiayu Sun. An anomaly-introduced learning method for abnormal event detection. *Multimedia Tools and Applications*, 77(22):29573–29588, Nov 2018.

[21] T. Hospedales, S. Gong, and T. Xiang. A markov clustering topic model for mining behaviour in video. In *ICCV*, pages 1165–1172, Sep. 2009.

[22] Jing Huo, Yang Gao, Wanqi Yang, and Hujun Yin. Abnormal event detection via multi-instance dictionary learning. In Hujun Yin, José A. F. Costa, and Guilherme Barreto, editors, *Intelligent Data Engineering and Automated Learning - IDEAL 2012*, pages 76–83, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[23] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, pages 448–456, 2015.

[24] Dinesh Jayaraman and Kristen Grauman. Slow and steady feature analysis: higher order temporal coherence in video. In *CVPR*, pages 3852–3861, 2016.

[25] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. In *ACM Multimedia*, MM '14, pages 675–678, New York, NY, USA, 2014. ACM.

[26] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. Large-scale video classification with convolutional neural networks. In *CVPR*, pages 1725–1732, 2014.

[27] Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision? In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *NIPS*, pages 5574–5584. Curran Associates, Inc., 2017.

[28] J. Kim and K. Grauman. Observe locally, infer globally: A space-time mrf for detecting abnormal activities with incremental updates. In *CVPR*, pages 2921–2928, June 2009.

[29] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2017.

[30] L. Kratz and K. Nishino. Anomaly detection in extremely crowded scenes using spatio-temporal motion pattern models. In *CVPR*, pages 1446–1453, 2009.

[31] W. L, W. Liu, and S. Gao. A revisit of sparse coding based anomaly detection in stacked rnn framework. In *ICCV*, Oct. 2017.

[32] Samuli Laine and Timo Aila. Temporal ensembling for semi-supervised learning. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2017.

[33] J. Larsen, L. Nonboe, M. Hintz-Madsen, and L. K. Hansen. Design of robust neural network classifiers. In *ICASSP*, 1998.

[34] N. Li, H. Guo, D. Xu, and X. Wu. Multi-scale analysis of contextual information within spatio-temporal video volumes for anomaly detection. In *The IEEE Conference on Image Processing (ICIP)*, pages 2363–2367, Oct. 2014.

[35] W. Li, V. Mahadevan, and N. Vasconcelos. Anomaly detection and localization in crowded scenes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36:18–32, 2014.

[36] Y. Li, D. Tarlow, M. Brockschmidt, and R. Zemel. Gated graph sequence neural networks. In *ICLR*, 2015.

[37] Y. Li, J. Yang, Y. Song, L. Cao, J. Luo, and J. Li. Learning from noisy labels with distillation. In *ICCV*, 2017.

[38] Tianwei Lin, Xu Zhao, and Zheng Shou. Single shot temporal action detection. In *ACM Multimedia*, pages 988–996, 2017.

[39] Guoqing Liu, Jianxin Wu, and Z-H Zhou. Key instance detection in multi-instance learning. In *Asian Conference on Machine Learning*, pages 253–268, 2012.

[40] W. Liu, W. Luo, D. Lian, and S. Gao. Future frame prediction for anomaly detection a new baseline. In *CVPR*, June 2018.

[41] C. Lu, J. Shi, and J. Jia. Abnormal event detection at 150 fps in matlab. In *CVPR*, pages 2720–2727, Dec. 2013.

[42] W. Luo, W. Liu, and S. Gao. Remembering history with convolutional lstm for anomaly detection. In *IEEE International Conference on Multimedia and Expo*, pages 439–444, July.

[43] Weixin Luo, Wen Liu, and Shenghua Gao. A revisit of sparse coding based anomaly detection in stacked rnn framework. In *ICCV*, Oct 2017.

[44] Vijay Mahadevan, Weixin Li, Viral Bhalodia, and Nuno Vasconcelos. Anomaly detection in crowded scenes. In *CVPR*, pages 1975–1981. IEEE, 2010.

[45] R. Mehran, A. Oyama, and M. Shah. Abnormal crowd behavior detection using social force model. In *CVPR*, pages 935–942, June 2009.

[46] Hossein Mobahi, Ronan Collobert, and Jason Weston. Deep learning from temporal coherence in video. In *NIPS*, pages 737–744. ACM, 2009.

[47] S. Mohammadi, A. Perina, H. Kiani, and M. Vittorio. Angry crowds: Detecting violent events in videos. In *ECCV*, pages 3–18, Oct. 2016.

[48] N. Natarajan, I. S. Dhillon, P. Ravikumar, and A. Tewari. Learning with noisy labels. In *NIPS*, 2013.

[49] N.Li, X. Wu, H. Guo, D. Xu, Y. O, and Y. Chen. Anomaly detection in video surveillance via gaussian process. *International Journal of Pattern Recognition and Artificial Intelligence*, 29:1555011, 2015.

[50] M. Ou, P. Cui, J. Pei, Z. Zhang, and W. Zhu. Asymmetric transitivity preserving graph embedding. In *KDD*, 2016.

[51] G. Patrini, A. Rozza, A. Krishna Menon, R. Nock, and L. Qu. Making deep neural networks robust to label noise: A loss correction approach. In *CVPR*, 2017.

[52] Joseph J. Pfeiffer, III, Sebastian Moreno, Timothy La Fond, Jennifer Neville, and Brian Gallagher. Attributed graph models: Modeling network structure with correlated attributes. In *ACM WWW*, pages 831–842, 2014.

[53] Zhaofan Qiu, Ting Yao, and Tao Mei. Learning spatio-temporal representation with pseudo-3d residual networks. In *ICCV*, pages 5534–5542. IEEE, 2017.

[54] Sivan Sabato and Naftali Tishby. Multi-instance learning with any hypothesis class. *Journal of Machine Learning Research*, 13(1):2999–3039, Oct. 2012.

[55] Clayton Scott, Gilles Blanchard, and Gregory Handy. Classification with asymmetric label noise: Consistency and maximal denoising. In *Proceedings of the 26th Annual Conference on Learning Theory*, volume 30 of *Proceedings of Machine Learning Research*, pages 489–511, Princeton, NJ, USA, 12–14 Jun 2013. PMLR.

[56] Zheng Shou, Hang Gao, Lei Zhang, Kazuyuki Miyazawa, and Shih-Fu Chang. Autoloc: Weakly-supervised temporal action localization in untrimmed videos. In *ECCV*, 2018.

[57] K. Simonyan and A. Zisserman. Two-stream convolutional networks for action recognition in videos. In *NIPS*, pages 568–576, 2014.

[58] Waqas Sultani, Chen Chen, and Mubarak Shah. Real-world anomaly detection in surveillance videos. In *CVPR*, June 2018.

[59] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In *CVPR*, pages 4489–4497, 2015.

[60] A. Vahdat. Toward robustness against label noise in training deep discriminative neural networks. In *NIPS*, 2017.

[61] H. Wang and C. Schmid. Action recognition with improved trajectories. In *CVPR*, pages 3551–3558, 2013.

[62] Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Van Gool. Temporal segment networks for action recognition in videos. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1, 2018.

[63] Xiaolong Wang and Abhinav Gupta. Videos as space-time region graphs. In *ECCV*, September 2018.

[64] Laurenz Wiskott and Terrence J Sejnowski. Slow feature analysis: Unsupervised learning of invariances. *Neural computation*, 14(4):715–770, 2002.

[65] S. Wu, B. E. Moore, and M. Shah. Chaotic invariants of lagrangian particle trajectories for anomaly detection in crowded scenes. In *CVPR*, pages 2054–2060, June 2010.

[66] Dan Xu, Yan Yan, Elisa Ricci, and Nicu Sebe. Detecting anomalous events in videos by learning deep representations of appearance and motion. *Computer Vision and Image Understanding*, 156:117 – 127, 2017.

[67] B. Zhao, F. Li, and E. P. Xing. Online detection of unusual events in videos via dynamic sparse coding. In *CVPR*, pages 3313–3320, June 2011.

[68] Y. Zhao, Y. Xiong, L. Wang, Z. Wu, X. Tang, and D. Lin. Temporal action detection with structured segment networks. In *ICCV*, Oct. 2017.

[69] Jia-Xing Zhong, Nannan Li, Weijie Kong, Tao Zhang, Thomas H. Li, and Ge Li. Step-by-step erasion, one-by-one collection: A weakly supervised temporal action detector. In *ACM Multimedia*, pages 35–44, 2018.

Graph Convolutional Label Noise Cleaner: Train a Plug-and-play Action Classifier for Anomaly Detection Supplementary Materials (Appendix)

Jia-Xing Zhong[1,2] Nannan Li[3,1,2] Weijie Kong[1,2] Shan Liu[4] Thomas H. Li[1] Ge Li ✉[1,2]
[1]School of Electronic and Computer Engineering, Peking University [2]Peng Cheng Laboratory
[3]Institute of Intelligent Video Audio Technology, Longgang Shenzhen [4]Tencent America
jxzhong@pku.edu.cn lnnsiat@gmail.com weijie.kong@pku.edu.cn
shanl@tencent.com tli@aiit.org.cn geli@ece.pku.edu.cn

1. Processing Speed in the Test Phase

	Input Size (Pixel)	Speed (FPS)
C3D	112×112	123.08
TSN-RGB	224×224	30.96
TSN-Optical Flow	224×224	150.15

Table 1: *Testing speed (FPS) of our models on a Titan-XP GPU.* Note that the reported result includes all pre-processing operations, such as resizing, 10-crop oversampling, zero-centering, *etc*.

Our approach of directly utilizing action classifiers for anomaly detection has great computational efficiency. As shown in Table 1, we report the frame per second (FPS) performance of the two types of action classifiers. Although the time-consuming pre-processes (*e.g*., 10-crop oversampling) are taken into consideration, the three action classifiers still have the real-time or even the super real-time performance.

2. Implementation of Label Noise Cleaner

At the first cleaning step, we select the 30% and 60% highest-confidence snippets as H for two-stream and C3D networks respectively if not specified, and increase the cardinality of H by 30% at each step. To learn an unbiased model, we also include normal videos in training data. To generate the label assignments of action classifiers, we concentrate the output probability into a single anomaly category with a min-max normalization. The output dimensions of the first two fully connected layers are 512 and 128 respectively, at the 60% dropout [10] rate. Both the graph modules have two convolutional layers: a 32-unit hidden layer activated by ReLu and the last 1-unit output layer. Due to the limited memory of GPUs, we at most sample 1,600 high-confidence snippets with not more than 8 neighbours respectively in a video. We implement our noise cleaner upon Pytorch [8] with the following hyper-parameters: $base\_learning\_rate = 0.0001$, $momentum = 0.9$ and $weight\_decay = 0.0005$. In preliminary experiments, we observe that three iterations are sufficient in most cases. Therefore we repeat the alternate optimization until the 3^{rd} step and compare the last (not always the best) results with other methods.

3. More Comparisons on UCSD-Peds

Several unary-classification works in 2018 also conduct experiments on *UCSD-Peds*. As shown in Table 3 and the main body of our paper, their default implementations are not directly comparable with ours because of different data splits. *For some open-source works, we hereby reproduce experiments on the data split in [1] as ours,* while the results in their original papers are also provided within square parentheses "[]" for reference as reported in Table 2. Since *UCF-Crime* is released at Github on June 10^{th} 2018 lately, except the official reference [11] and its comparisons, neither public reporting of results nor source codes can be found, and we hope that our work can fill in the blanks.

4. Vectorized Feature Similarity Module

Following the main body of our paper, we denote the feature similarity graph as $F = (V, E, \mathbf{X})$, where V is the vertex set, E is the edge set, and $\mathbf{X}$ is the attribute of vertexes. In particular, V is a video, E describes the feature similarity amongst snippets, and $\mathbf{X} \in \mathbb{R}^{\mathbf{N} \times \mathbf{d}}$ represents the d-dimensional feature of these N snippets. The adjacency matrix $\mathbf{A} \in \mathbb{R}^{N \times N}$ of F is defined as:

$$\mathbf{A}_{(i,j)} = \exp(\mathbf{X}_i \cdot \mathbf{X}_j - max(\mathbf{X}_i \cdot \mathbf{X}))\,, \qquad (1)$$

Method	Publication	AUC (%)
Unary-classification Paradigm		
...	...	...
TCP [9]	WACV 2018	No source codes [88.4]
Frame Prediction [6]	CVPR 2018	**92.6 ± 1.1 [95.4]**
C2ST [7]	BMVC 2018	81.4 ± 2.8 [87.5]
Binary-classification Paradigm		
AL [1]	J-Mult. Nov. 2018	90.1
Ours-TSN$^{Gray-scale}$	–	**93.2 ± 2.3**
Ours-TSN^{OpticalFlow}	–	92.8 ± 1.6

Table 2: *Comparison on UCSD-Peds in 2018.* The results of their original papers under data split [5] are reported within "[]".

Splitting Approach	Train		Test
	Normal	Abnormal	
Following [5]	16	0	12
Following [1]	4	6	18

Table 3: *Difference in splitting UCSD-Peds.* The random selection is repeated 10 times in [1].

where the element $\mathbf{A}_{(i,j)}$ measures the feature similarly between the i^{th} and j^{th} snippets. Here is an equivalent vectorization of Equation 1:

$$\mathbf{A} = \exp(\mathbf{X}\mathbf{X}^T - torch.max(\mathbf{X}\mathbf{X}^T, dim = 1)), \quad (2)$$

where the $torch.max$ function takes the maximum value over dimension 1.

The nearby vertexes are driven to have the same anomaly label via the graph-Laplacian operation approximated with a renormalization trick [3]:

$$\widehat{\mathbf{A}} = \widetilde{\mathbf{D}}^{-\frac{1}{2}}\widetilde{\mathbf{A}}\widetilde{\mathbf{D}}^{-\frac{1}{2}}, \quad (3)$$

where the self-loop adjacency matrix $\widetilde{\mathbf{A}} = \mathbf{A} + \mathbf{I_n}$, and $\mathbf{I_n} \in \mathbb{R}^{N\times N}$ is the identity matrix; $\widetilde{\mathbf{D}}$ is the corresponding degree matrix:

$$\widetilde{\mathbf{D}}_{(i,i)} = \sum_j \widetilde{\mathbf{A}}_{(i,j)}. \quad (4)$$

The vectorization of Equation 4 is implemented with the vectorized summation and the broadcasting diagonal functions of Pytorch:

$$\widetilde{\mathbf{D}} = torch.diag(torch.sum(\widetilde{\mathbf{A}}, dim = 1)). \quad (5)$$

Finally, the output $\mathbf{H}$ of a feature similarity graph module layer is computed as:

$$\mathbf{H} = \sigma(\widehat{\mathbf{A}}\mathbf{X}\mathbf{W}), \quad (6)$$

where $\mathbf{W}$ is a trainable parametric matrix, and σ is an activation function.

Since the whole computational procedure is differentiable, our feature similarity graph module can be trained in an end-to-end fashion. Therefore, neural networks are capable of seamlessly incorporating the single or multiple stacked modules. The temporal similarity module can be also rewritten as its corresponding vectorized implementation in a similar manner.

5. Details of Indirectly Supervised Loss Term

Our indirectly supervised term of the loss function can be viewed as a temporal ensembling strategy [4]. The pseudo code is shown in Algorithm 1. In practice, we set γ as 0.5 in all of the experiments. Since we have already obtained a set of rough predictions from the action classifier, the "cool start" initialization and the bias correction of the original temporal ensembling method [4] are not required as illustrated on the 1^{st} and the 8^{th} statements.

6. Reorganization of ShanghaiTech

	Training Set	Test Set	Total
Normal Videos	175	155	330
Anomaly Videos	63	44	107
Total	238	199	437

Table 4: *The number of videos on our reorganized ShanghaiTech.*

In total, there are 437 videos on ShanghaiTech. As shown in Table 4, we split the data into two subsets: the training set is made up of 238 videos, and the testing set contains 199 videos. In each scene, the numbers of normal and anomaly videos w.r.t. the two subsets are depicted in Figure 1 and Figure 2, respectively. The new

data split is available at https://github.com/jx-zhong-for-academic-purpose/GCN-Anomaly-Detection.

7. Discuss the Formulation

Following the reviewer's suggestion, we discuss our noisy-labeled problem formulation and the EM-like optimization mechanism under this formulation in more detail.

7.1. Concept: MIL vs Noisy-labeled Learning

Conceptually, the two formulations mainly differ in their *emphases*. Given a positive bag $Y = 1$, the MIL usually focuses on *positive instances* $y_i = 1$, whereas the noisy-labeled training pays attention to *noisy labels* $y_i = 0$ and the remaining ones are $y_i = 1$. The two conceptions are complementary and have transformational relations.

7.2. Practice: EM-like MIL vs Ours

Practically, in terms of *selection criteria on "seed examples"*, the EM-like MIL focuses on the *most-likely positive instances*, while our noisy-labeled optimization prefers the *most-likely reliable predictions*. Take the three MIL models the reviewer mentioned for examples. If the 10-crop prediction of a snippet within an anomalous video is {0.2, 0.2, ..., 0.2}, He *et al*. [1] will not update their "anchor dictionary" with it for its low anomaly score *(mean value=0.2)*, Hou *et al*. [2] will exclude it because it is "non-discriminative" (without "the same label" as the corresponding video), Zhang *et al*. [12] will neglect it since their E-step is to seek the most "responsible" instance to the bag annotation, but we will select it to supervise our GCN because it is highly certain and noiseless *(predictive variance=0)*.

7.3. Terminology: EM-like vs EM-based

As pointed out in the main body of this paper, our updating method is "EM-like" instead of "EM-based". The resemblance between our optimization mechanism and the EM-based approach is that they both alternately repeat update-and-fix processes. However, our method is not "EM-based" since we do not explicitly estimate mathematical expectation in the training process.

References

[1] Chengkun He, Jie Shao, and Jiayu Sun. An anomaly-introduced learning method for abnormal event detection. *Multimedia Tools and Applications*, 77(22):29573–29588, Nov 2018. 1, 2, 3

[2] L. Hou, D. Samaras, T. M. Kurc, Y. Gao, J. E. Davis, and J. H. Saltz. Patch-based convolutional neural network for whole slide tissue image classification. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2424–2433, June 2016. 3

[3] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2017. 2

[4] Samuli Matias Laine and Timo Oskari Aila. Temporal ensembling for semi-supervised learning, Apr. 12 2018. US Patent App. 15/721,433. 2

[5] W. Li, V. Mahadevan, and N. Vasconcelos. Anomaly detection and localization in crowded scenes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(1):18–32, 2013. 2

[6] W. Liu, W. Luo, D. Lian, and S. Gao. Future frame prediction for anomaly detection a new baseline. In *CVPR*, June 2018. 2

[7] Yusha Liu, Chun-Liang Li, and Barnabás Póczos. Classifier two-sample test for video anomaly detections. In *BMVC*, 2018. 2

[8] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017. 1

[9] M. Ravanbakhsh, M. Nabi, H. Mousavi, E. Sangineto, and N. Sebe. Plug-and-play cnn for crowd motion analysis: An application in abnormal event detection. In *WACV*, 2018. 2

[10] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15:1929–1958, 2014. 1

[11] Waqas Sultani, Chen Chen, and Mubarak Shah. Real-world anomaly detection in surveillance videos. In *CVPR*, June 2018. 1

[12] Qi Zhang and Sally A Goldman. Em-dd: An improved multiple-instance learning technique. In *Advances in neural information processing systems*, pages 1073–1080, 2002. 3

Algorithm 1 *Indirectly Supervised Loss Term.*
Note that the practical computational processes are incrementally implemented, while in this pseudo code all of them are calculated from the 1^{st} epoch for clarity.

Input:

$V = \{v_i\}_{i=1}^{N}$: a video with N snippets
$\widetilde{Y} = \{\widetilde{y}_i\}_{i=1}^{N}$: the rough snippet-wise anomaly probabilities from the last action classifier
$p_\theta(v_i)$: the GCN predictions of video clips v_i with trainable parameters θ
$\alpha(v_i)$: the stochastic augmentation (such as dropout and random cropping) function of input snippets v_i
γ: a hyper-parametric discount factor within the range of $(0, 1)$

Output:

$\mathcal{L}_I^j$: the indirectly supervised loss at the j^{th} epoch

Initialize the smooth target $\overline{p}_{i\in1,2,..,N} = \widetilde{y}_{i\in1,2,..,N}$
repeat
  Initialize the epoch counter $j = 0$
  for each video V in the training set **do**
    Obtain the GCN predictions of augmented snippets: $p_i = p_\theta(\alpha(v_i))$
    Compute the loss under indirect supervision: $\mathcal{L}_I^j = \frac{1}{N}\sum_{i=1}^{N}|p_i - \overline{p}_i|$
    Optimize the parameters θ of the GCN
    Update the smooth target: $\overline{p}_{i\in1,2,..,N} = \gamma\overline{p}_{i\in1,2,..,N} + (1-\gamma)p_{i\in1,2,..,N}$
  Update the epoch counter: $j = j + 1$
until j == current epoch number

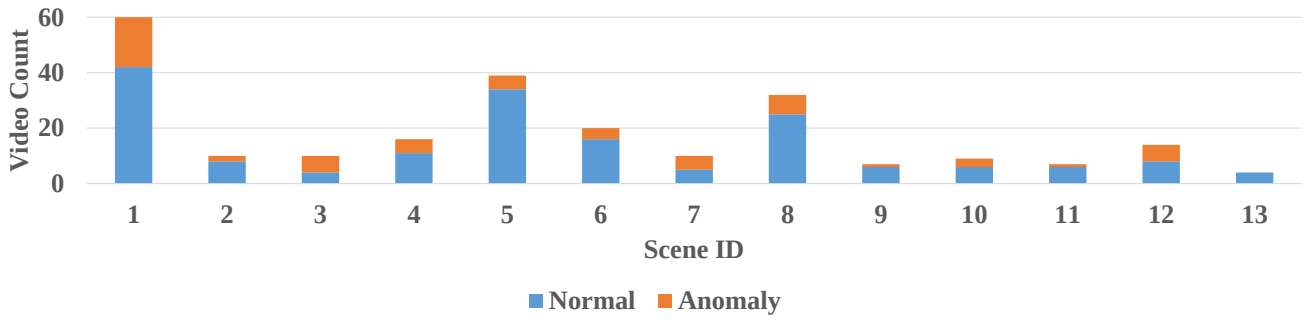


Figure 1: *Training set on the reorganization of ShanghaiTech.*

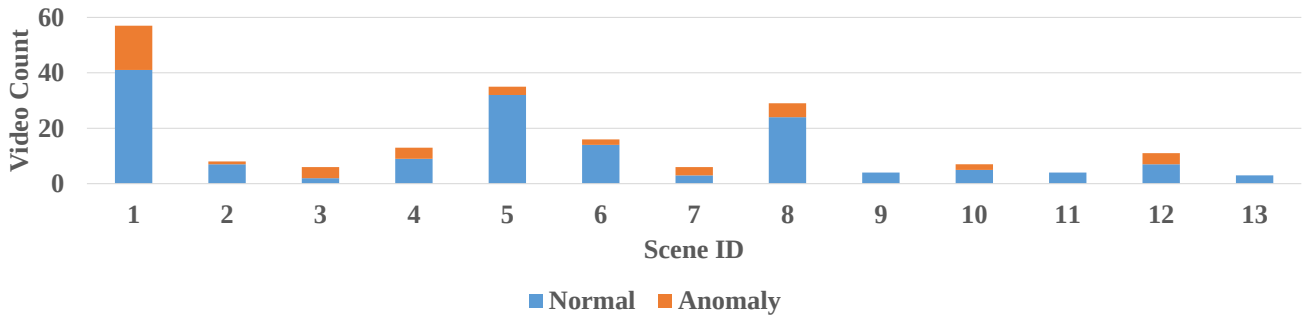


Figure 2: *Testing set on the reorganization of ShanghaiTech.*