https://doi.org/10.2352/EI.2023.35.9.IPAS-286

Detection of Object Throwing Behavior in Surveillance Videos

Ivo P.C. Kersten, Erkut Akdag†, Egor Bondarev, and Peter H.N. de With
Eindhoven University of Technology, Department of Electrical Engineering, 5612 AP Eindhoven, The Netherlands
†*Corresponding author : e.akdag@tue.nl*

Abstract

Anomalous behavior detection is a challenging research area within computer vision. Progress in this area enables automated detection of dangerous behavior using surveillance camera feeds. A dangerous behavior that is often overlooked in other research is the throwing action in traffic flow, which is one of the unique requirements of our Smart City project to enhance public safety. This paper proposes a solution for throwing action detection in surveillance videos using deep learning. At present, datasets for throwing actions are not publicly available. To address the use-case of our Smart City project, we first generate the novel public 'Throwing Action' dataset, consisting of 271 videos of throwing actions performed by traffic participants, such as pedestrians, bicyclists, and car drivers, and 130 normal videos without throwing actions. Second, we compare the performance of different feature extractors for our anomaly detection method on the UCF-Crime and Throwing-Action datasets. The explored feature extractors are the Convolutional 3D (C3D) network, the Inflated 3D ConvNet (I3D) network, and the Multi-Fiber Network (MFNet). Finally, the performance of the anomaly detection algorithm is improved by applying the Adam optimizer instead of Adadelta, and proposing a mean normal loss function that covers the multitude of normal situations in traffic. Both aspects yield better anomaly detection performance. Besides this, the proposed mean normal loss function also lowers the false alarm rate on the combined dataset. The experimental results reach an area under the ROC curve of 86.10 for the Throwing-Action dataset, and 80.13 on the combined dataset, respectively.

Index Terms— Anomaly Detection, Throwing Actions, UCF-Crime, CNN

Introduction

Surveillance cameras are increasingly deployed to monitor public spaces to enhance safety. The presence of surveillance cameras deters people from misbehaving and accelerates the detection of unsafe behavior in traffic. However, the increasing amount of surveillance footage processed by control-room operators adversely impacts the actual efficiency of these cameras. The human operators try to follow many camera streams at once, which results in a higher likelihood of missing anomaly events. Anomaly events are defined as deviations from the normal patterns in traffic flow. Such events can range from any type of traffic accident up to suspicious behavior of people. Although anomalies in traffic flow occur rarely, they have a significant impact on public safety. Therefore, detecting these anomalies should be automated to ensure maximum efficiency of a surveillance system and improve the detection of anomaly events to sustain public safety.

The automation of anomaly detection is a topic that has recently seen an increase in attention from the research community, where a focus is placed on detecting a limited number of severe anomalies. However, certain types of anomalies have received hardly any attention from the research community, even though they can result in dangerous situations. One such anomaly is the action of throwing objects into the surroundings by car drivers or other traffic participants. Such throwing activities can have crucial consequences, from disrupting the traffic flow to creating dangerous situations for all traffic participants. Given the impact of this action, it is worth noting that we have found no research work that aims at the detection of throwing anomalies. For this reason we have constructed a dataset with throwing actions by different traffic participants. Furthermore, this study explores how to apply the anomaly-analysis approaches to the problem of detecting throwing actions and how to merge our throwing action dataset into existing anomaly detection datasets. The ultimate goal of our research is to enable automated detection of dangerous behavior using surveillance camera feeds. A dangerous behavior has been neglected often in other research and contains multiple dimensions, where throwing actions in traffic flow is one of the unique requirements and an important use case in Smart City projects.

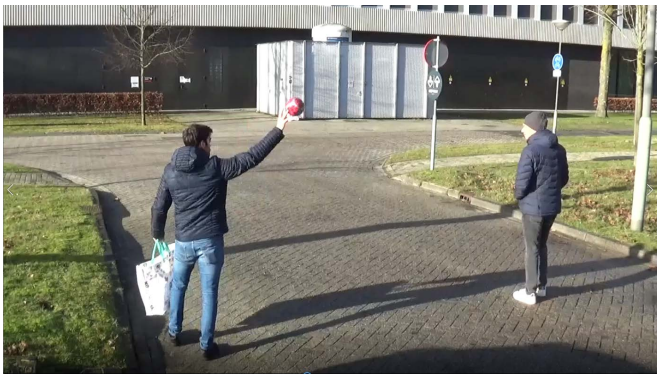

Figure 1. *Example frame from one of the videos in the proposed generated Throwing-Action dataset, where a pedestrian is throwing an object at another pedestrian.*

This paper is organized as follows. First, the section on Related Work provides an overview of throwing action detection and general anomaly detection in the literature. Next, in the Proposed Novel Dataset section, we introduce the novel throwing action dataset. The Methodology section presents our proposed method and contributions. Then we evaluate experimental results in the Experiments, while in the False Alarm Analysis section, the performance of the proposed anomaly detection method is analyzed with respect to the false alarm rate and qualitative results. Finally, the paper is concluded in the Conclusion section.

Related Work

This section presents the literature overview on methods for throwing action detection and on supervised, unsupervised, and semi-supervised anomaly detection methods.

A. Throwing Action Detection

Related to throwing action detection, previous studies mainly focus on the detection of the thrown object itself instead of the action. In [5], a modified version of YOLOv3 [17] is proposed to detect particular objects when they are thrown out of a car window. This approach relies on the recognition of specific small objects, which is difficult when dealing with low-resolution video footage or throwing actions are performed far away from the camera.

Other studies in [4] and [18] concentrate on the trajectories of moving objects. The authors fit parabolic trajectories into the potential trajectories. In other words, any object that is found to follow a parabolic trajectory is deemed as a thrown object. This approach requires long trajectories of thrown objects to reliably fit a parabolic trajectory. In practice, objects are often thrown horizontally or downwards, which results in a short trajectory that renders this method inefficient and only partially addressing the case.

B. Anomaly Detection

Anomaly detection is a challenging computer vision problem aiming to detect rare events from a video stream. Different approaches to this problem can be split into unsupervised, supervised, and semi-supervised categories.

1) Unsupervised Anomaly Detection

The most popular approach is unsupervised anomaly detection, where the training set contains only normal sequences. This approach is employed in many papers, such as [2, 6, 10, 14, 15, 19, 21, 27]. During the training stage, a model of normal behavior is constructed. This model can properly characterize normal data in the testing stage, whereas it cannot characterize anomalous data. Applying this method is advantageous when the exact nature of the target anomalies is unknown or uncertain. A drawback of unsupervised anomaly detection is that creating a model capable of exactly capturing all possible normal behaviors is complicated. In other words, similar behaviors can be often both anomalous or normal, depending on the context. For example, a car driving down the right side of the road is considered normal behavior, while driving down the wrong side of the road is not.

2) Supervised Anomaly Detection

The second type is supervised training to learn anomalous behavior. Several authors [13, 16, 26] use supervised anomaly detection methods to predict the presence of anomalies only at the temporal level, while others [11, 27] also locate the anomalies at the spatial level. In contrast to the unsupervised methods that typically rely on a reconstruction error to find abnormal inputs, supervised learning generally learns to predict an anomaly score directly from the input data. An advantage of the supervised anomaly detection methods is that they often outperform unsupervised techniques for the specific anomalies for which they are trained. One of the drawbacks is the ambiguity of where specifically an anomaly begins and ends. The second drawback is that these methods can only detect the anomalies existing in the dataset, furthermore, annotating the training data is labor-intensive and expensive, as anomaly locations should be specified in every video frame or segment.

3) Semi-supervised Anomaly Detection

The last type of anomaly detection is semi-supervised anomaly detection. This method for anomaly detection is first introduced in [23] and is further expanded in [12, 24]. In this approach, the training data contains both videos of normal and anomalous events, however, the data is annotated on a per-video basis. It is significantly faster to annotate data in this way, compared to labeling at individual frames or areas in each frame. This annotation brings a major advantage over the fully-supervised anomaly detection methods. A disadvantage of this approach is that it only learns to recognize anomalies that occur in the training data.

As can be derived, previous studies mainly focus on object or trajectory detection, instead of considering the throwing action detection as an anomaly detection use case. Therefore, we consider these throwing actions as anomalies, and generate a dataset of throwing actions performed by different road users in real traffic flow. We opted for a semi-supervised anomaly detection method, as opposed to a supervised anomaly detection method, because of its advantages in requiring less labeled data, while still providing a higher accuracy than the unsupervised methods.

Proposed Novel Dataset

Currently, datasets are not available from literature to train a throwing anomaly detector. Therefore, we have generated a novel dataset, including the throwing anomalies in six different outdoor categories to address the anomaly use case from the Smart City project. The throwing anomalies are split into classes based on the acting traffic participant (car, bicycle, pedestrian) performing the throw, and whether or not the thrown object is directed towards another traffic participant. If the throw is directed at another traffic participant, it is called a 'dangerous' throw, otherwise a 'safe' throw. The generated dataset includes six throwing anomaly classes as described in Table 1. All videos have a resolution of 320×240 pixels. One of the challenges in our dataset is that throwing anomaly videos can contain multiple throwing actions up to a maximum of ten.

The objects used to create the throwing anomaly dataset are selected to have a large diversity in object shape, size, and color. Additionally, some objects maintain their shape during a throwing action, such as a football, while others become deformed and change their shape during the throwing movement, such as a sweater or plastic bag. Overall, the generated dataset consists of 130 normal videos without throwing anomalies and 271 anomalous videos divided over all six anomaly classes, titled as the *"Throwing-Action"* dataset throughout the paper. Table 2 provides the number of videos of each anomaly class in the dataset.

A. Annotation

In the training set, each video is labeled as either normal or anomalous at the video sequence level, while each video is annotated at the frame level in the testing set. In other words, the start and end frames of each throwing action are provided for the test set. While the start frame of an anomaly action is defined as

Class	Definition	Examples of video
Bicycle Dangerous	Object is thrown by a bicyclist towards another traffic participant	
Bicycle Safe	Object is throwing by a bicyclist onto the ground	
Car Dangerous	Object is thrown out of a car window towards another traffic participant	
Car Safe	Object is thrown out of a car window onto the ground	
Pedestrian Dangerous	Object is thrown by a pedestrian towards another traffic participant	
Pedestrian Safe	Object is thrown by a pedestrian onto the ground	
Normal	No throwing actions occur but some of the traffic participants do occur in the video	

Table 1. Definition and examples of each video category in the Throwing-Action dataset.

	Throwing-Action		
	Training	Testing	Total
Bicycle Dangerous	42	21	63
Bicycle Safe	20	10	30
Car Dangerous	38	19	57
Car Safe	22	11	33
Pedestrian Dangerous	38	20	58
Pedestrian Safe	20	10	30
Normal	87	43	130

Table 2. Distribution of the number of videos over the various classes for the Throwing-Action dataset.

the first frame in which the object starts moving, the end frame of it is considered as the first frame where the object touches the ground or is occluded.

B. Training and Testing Sets

The generated "Throwing-Action" dataset is divided into a training and testing set. The training set consists of 87 normal videos and 180 anomalous videos, while the testing set contains the other 43 normal videos and 91 anomalous videos. Table 2 shows the number of videos for each class in both subsets.

Methodology

A. Baseline Method

In this paper, we adopt an anomaly detection method inspired by [23] as a baseline model, based upon the implementation of [9]. Figure 2 depicts a diagram of the anomaly detection method. During the training process, input videos are split into 32 non-overlapping segments of equal length. These segments can be considered as instances in a bag, where the bag represents the video as a whole. To describe these segments, feature vectors are extracted from every 16-frame video clip within the instance, using a pre-trained feature extractor network. To represent the instance, we take the average of all 16-frame video clip features within that segment. Next, the instance-wise feature vectors are supplied to a feed-forward network, which aims to predict the likelihood that the input instance contains an anomaly, resulting in the representation of each instance in the bag by the predicted anomaly score. A multiple instance learning ranking loss ensures that one anomalous and one normal video are used for each training step. The notations $\mathscr{V}_a$ and $\mathscr{V}_n$ represent an anomalous and a normal video, respectively, while $\mathscr{A}_a^i$ and $\mathscr{A}_n^i$ represent the predicted anomaly score of the i-th segment of an anomalous and a normal video, respectively. The multiple-instance learning ranking loss function can then be expressed by:

$$\mathscr{L}_{\text{orig}}(\mathscr{V}_a, \mathscr{V}_n) = \max\,(\,0\,,\; 1 - \max_{i \in \mathscr{V}_a} f(\mathscr{A}_a^i) + \max_{i \in \mathscr{V}_n} f(\mathscr{A}_n^i)\,). \quad (1)$$

This loss function forces the maximum anomaly score of the normal video towards zero, while simultaneously increasing the maximum anomaly score of the anomalous video towards unity. The assumption is made that the segment with the highest anomaly score is the true anomaly.

Next to this multiple-instance learning ranking loss component, three more components are added to the loss in order to ensure temporal smoothness, sparsity and small model weights with scaling factor λ_1, λ_2 and λ_3, respectively. These additions result in the objective function given by

$$\begin{aligned}\mathscr{L}_{\text{obj}} = \mathscr{L}_{\text{orig}}(\mathscr{V}_a, \mathscr{V}_n) \;+\; \lambda_1 \sum_{i}^{(n-1)} (f(\mathscr{A}_a^i) - f(\mathscr{A}_a^{i+1}))^2 \\ +\; \lambda_2 \sum_{i}^{n} f(\mathscr{A}_a^i) \;+\; \lambda_3 \|\mathscr{W}\|_F\,,\end{aligned} \quad (2)$$

where $\mathscr{W}$ represents the model weights, and n the number of segments into which a video is split.

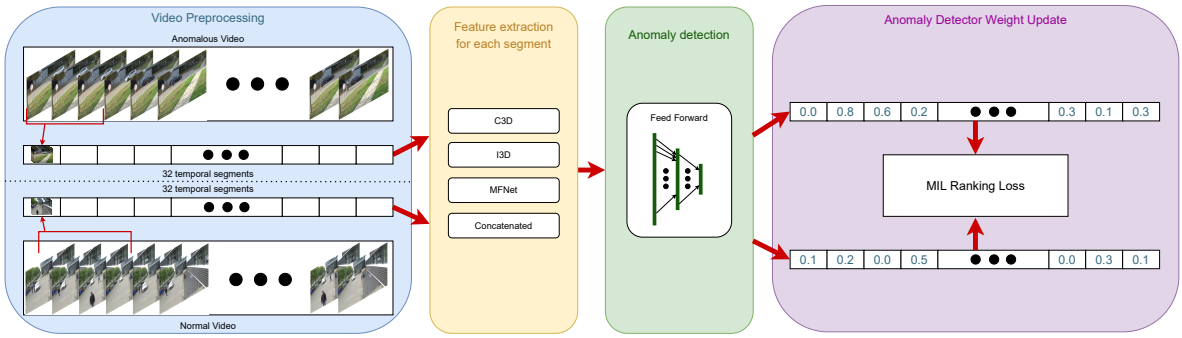

Figure 2. *Flow diagram of the proposed methodology. Each video is split into 32 temporal segments. The videos are represented as a bag, and the segments as instances within this bag. Features are then extracted from each segment by the pre-trained C3D, I3D, or MFNet feature extraction networks. Next, these features are provided with an anomaly score by a fully connected neural network, resulting in one anomaly score for every instance in the bag. The network uses a multiple instance learning ranking loss for training.*

B. Variations on the Baseline Method

We experiment with different feature extractors within our methodology and forms of our loss functions, which both are discussed briefly below. The paper that inspired our proposal applied the C3D network [25] only, whereas the further investigated feature extractors are the I3D network [1], MFNet network [3], and features concatenated from the aforementioned extractors.

C3D network: The C3D network is originally proposed in [25] and is one of the first applications of a three-dimensional convolutional neural network (3D-CNN) for supervised video action classification. We extract the output of the first fully connected layer as our features. In this work, the C3D feature extractor is pre-trained on the Sports1M dataset [8].

I3D network: The I3D network [1] is based on the Inception-V1 [7] detector. We take the output of the average-pooling layer as our computed features, resulting in a 1024-dimensional feature vector. Here, the I3D network is pre-trained on two different datasets, the Charades dataset [20] and the Kinetics dataset [1]. These feature extraction backbones are referred to as I3D-charades and I3D-kinetics, respectively, throughout the paper.

MFNet network: The MFNet network [3] is used to extract features from the input videos. In literature, this network is not yet applied for the purpose of anomaly detection in videos. The MFNet network achieves slightly better performance than I3D on action-recognition datasets, while at the same time, it requires up to ten times fewer computations according to [3]. The MFNet feature extraction backbone is pre-trained on the UCF-101 dataset [22]. The MFNet features are represented by 6,144-dimensional vectors.

Concatenated features: Finally, we concatenate all the extracted features from the C3D, I3D-charades, I3D-kinetics and MFNet networks together, of which we construct a 12,288-dimensional feature vector. The anomaly detection model has access to a wider range of information, thereby improving anomaly detection performance with the help of these concatenated features. The downside of this approach is the increase in computational cost to execute all feature extraction networks.

In our method, the loss is based only on the maximum obtained anomaly score of any segment in a video, both for normal and anomalous videos. For an anomalous video, this makes sense because it could be possible that such a video only contains one anomalous segment. However, all segments should ideally have an anomaly score of zero for normal videos. Hence, we experiment with a loss function that takes the mean predicted anomaly score into account, as opposed to only the maximum. This loss is from now on referred to as the mean normal loss and described formally below. This change allows the model to learn from all normal segments at every training iteration, so that the mean normal loss is specified by

$$\mathscr{L}_{\text{mean-nl}}(\mathscr{V}_a, \mathscr{V}_n) = \max\left(0,\ 1 - \max_{i \in \mathscr{V}_a} f(\mathscr{A}_a^i) + \operatorname*{mean}_{i \in \mathscr{V}_n} f(\mathscr{A}_n^i)\right). \quad (3)$$

Finally, we apply different optimizers in order to compare their effects on the model performance. The baseline paper [23], uses the Adadelta optimizer, while we also apply the Adam optimizer.

Experiments

In this section, we perform experiments with our anomaly detection method described in the Methodology section on the UCF-Crime dataset and the generated Throwing-Action dataset to improve the anomaly detection performance. The primary metric used to compare anomaly detection performance between experiments is the area under the Receiver Operating Characteristic (ROC) curve. This metric is commonly used to evaluate the performance of anomaly detection methods and it gives insight into the achieved ratio between the true-positive rate and false-positive rate for the full range of anomaly thresholds.

A. Comparison of Adadelta and Adam Optimizers

Our first contribution is determining the appropriate optimizer for the training phase of all experimentation models. We train an anomaly detection model on the Throwing-Action training dataset for 100,000 iterations, using the Adadelta optimizer with a learning rate of 0.01, similarly to [23] and using the Adam optimizer with a learning rate of 0.0005. Figure 3 depicts the optimal ROC curves for each model.

As shown in Figure 3, Adam outperforms Adadelta by a significant margin. This difference indicates that the Adadelta training process has not reached an optimum after 100,000 iterations. Figure 4 shows the batch loss during training for both the Adadelta and Adam optimizers. From this graph, it can be

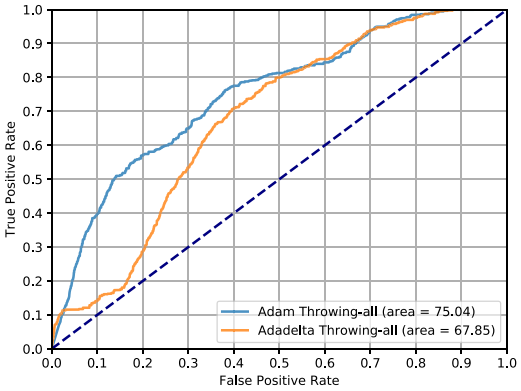

Figure 3. *Optimal ROC curves obtained on the testing set of the Throwing-Action dataset when the anomaly detection model is trained with the Adadelta and Adam optimizers.*

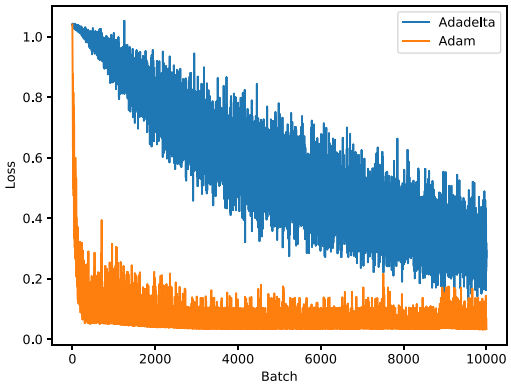

Figure 4. *Training loss curves of the anomaly detection model on the Throwing-Action dataset for Adadelta and Adam optimizers, showing the mean loss of every batch during training.*

observed that the learning process is significantly slower for the Adadelta optimizer. Therefore, the Adam optimizer is found more suitable to train anomaly detection models. Since our anomaly detection model trained with the Adam optimizer performs better and has a faster learning process, all models are from now on trained using the Adam optimizer with a learning rate of 0.0005.

B. Video Augmentation

Next, several approaches for data augmentation are explored to improve performance. Augmentation methods are applied to the videos of the Throwing-Action dataset to diversify the training data, which helps the model in generalizing to new cases. Furthermore, test-time augmentation (TTA) is explored for augmentation as well.

1) Training set augmentation

We have applied five augmentation methods to the original videos. The augmentation methods used are salt-and-pepper noise, where in total 3% of all sub-pixels are affected; inverting colors; horizontal flipping; independently scaling the color channels by a factor between 0.8 and 1.2; and finally, applying shear transformations. Visual examples of each augmentation output

	No Augment.	Augment.	Test Time Augment.
C3D	75.04	76.23	75.55
I3D-charades	78.71	80.83	83.15
I3D-kinetics	80.41	82.84	84.32
MFNet	**86.10**	83.63	83.89
Concatenated	85.29	**85.58**	**85.52**

Table 3. Maximum area under the ROC curve achieved on the Throwing-Action dataset for all feature extraction networks, when applying different augmentation methods (best scores in bold).

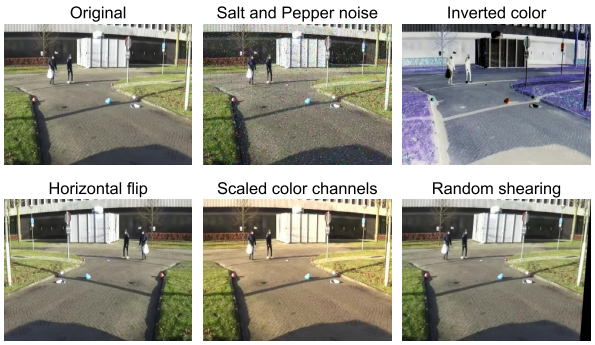

Figure 5. *Visual examples of each type of image augmentation applied to the Throwing-Action dataset. From left to right and top to bottom: original, (no augmentation), salt and pepper noise, inverted color, horizontal flip, independently scaled color channels and random shearing.*

are illustrated in Figure 5.

An anomaly detection model is trained on both the original Throwing-Action dataset and the augmented Throwing-Action dataset for each feature extraction network. Table 3 shows the largest area under the ROC curve obtained on the test set of the Throwing-Action dataset for all feature extractors. From this table, we can conjugate that the choice of the feature extraction has a significant impact on the quality of the resulting model. The C3D feature extraction network performs worst, while MFNet and the concatenated features perform best. Training with the concatenated features does not improve the performance of any individual feature extractor. The MFNet features contain all necessary information for detecting the anomalies, while the other feature extractors effectively only manage to extract a subset of this information. This is visible because MFNet network obtains the highest score, while the other ones and even the concatenated network score lower. Table 3 indicates that training on augmented data improves the obtained results for all feature extractors, except for MFNet. We obtain the best results for the MFNet feature extractor without augmentation while for concatenated features augmentation is needed. However, due to the significantly increased computational cost of the concatenated feature extractor compared to the MFNet feature extractor, the final model uses MFNet without data augmentation.

2) Test time augmentation (TTA)

Another technique that can improve the performance of the anomaly detection model is test time augmentation (TTA). The videos in the test set are augmented using the same augmenta-

	Exp. 1	Exp. 2	Exp. 3	Exp. 4
C3D	70.30	69.80	67.68	70.66
I3D-charades	78.36	77.95	79.80	77.64
I3D-kinetics	78.47	78.34	80.25	78.01
MFNet	76.76	72.12	75.19	71.50
Concatenated	**79.84**	79.16	**81.08**	78.86
Concatenated Mean normal loss	-	**80.13**	78.59	**80.31**

Table 2. Maximum area under the ROC curve achieved for different experiments. Exp 1: model trained and evaluated on UCF-Crime dataset. Exp 2, 3 and 4: model trained on UCF-Crime+Throwing dataset, evaluated on UCF-Crime+Throwing, Throwing-Action and UCF-Crime datasets, respectively.

tion methods as used for the training set, while predictions of the anomaly detection model are averaged over all versions of the same video. Table 3 shows the maximum area under the ROC curves when training the anomaly detection model on the augmented Throwing-Action dataset and testing with TTA. TTA further improves the results for both versions of the I3D feature extractors, but does not manage to improve results for other feature extractors. Furthermore, TTA requires augmented test data which makes the model more expensive at test time.

In conclusion, since the optimal area under the ROC curve is obtained by a model trained without any augmentation and training with augmentation is more computationally expensive, we have decided to leave out data augmentation in the remainder of this paper.

C. Experiments on UCF-Crime+Throwing Dataset

In addition to anomaly detection performance on the newly generated Throwing-Action dataset, we are interested in combining the new Throwing-Action dataset with the publicly available UCF-Crime [23] anomaly dataset. Therefore, we first evaluate the performance achieved on the UCF-Crime dataset for all feature extractors.

Table 4 shows the best maximum area under the ROC curve on the UCF-Crime dataset for all feature extraction backbones. It indicates that the performance for the C3D feature extraction is significantly lower than other feature extractors. Another observation from Table 4 is that concatenated features outperform any individual feature extraction backbone. For the UCF-Crime dataset, the different feature extractors provide different sets of information about the video, and concatenating these allows the detector model to consider more information. Furthermore, comparing the performance on the UCF-Crime dataset with the performance on the Throwing-Action dataset shown in Table 3 reveals that for each feature extraction network the performance on the UCF-Crime dataset is lower, leading to the conclusion that the UCF-Crime dataset is overall a more difficult dataset.

Next, we combine the UCF-Crime dataset with our Throwing-Action dataset. This new dataset is from now on referred to as UCF-Crime+Throwing. In order to compare the performance, an anomaly detection model is trained on the training subset of the UCF-Crime+Throwing dataset for each feature extraction backbone and evaluated on the UCF-Crime+Throwing, the Throwing-Action, and the UCF-Crime testing sets. Table 4 shows the maximum area under the ROC curve obtained on each test set for every feature extraction backbone. The results indicate that the performance on the UCF-Crime+Throwing dataset is generally similar to the performance on only the UCF-Crime dataset. The results show that it is possible to add throwing anomaly detection capabilities at only a small cost in the detection performance of other anomalies. Therefore, it is viable to integrate throwing anomaly detection into general anomaly detection systems.

The final experiment is concerned with the proposed mean normal loss function. Since this loss enables the model to learn from all normal segments at every training iteration, changing the loss function allows the model to better recognize normal sections of a video, which decreases the false positive rate and increases the area under the ROC curve. Table 4 summarizes the performance for this modified loss function when using concatenated features and training on the UCF-Crime+Throwing dataset. This model achieves the highest overall performance on the UCF-Crime+Throwing dataset, but at the cost of decreased performance on the Throwing-Action testing set.

D. Qualitative results

This section presents qualitative results of the proposed anomaly detection method. Figure 6 shows the output of the feed-forward network as a function of time, expressed by the frame number of several testing videos of the Throwing-Action dataset. This means that the y-axis shows the likelihood that a segment is anomalous, according to our anomaly detection method, and the x-axis shows the progression number of processed video frames. Furthermore, the ground-truth anomalous regions within each video are given as red colored areas. In Figure 6 (a)-(d), it can be observed that the performance is quite good for pedestrian and bicycle-related anomalies. However, several segments have a high predicted anomaly score, while being outside of the ground-truth anomalous areas. This problem occurs most often between two distinct anomalies or just after one anomaly. These segments contain relatively large person-arm motions after completing a throw, which may seem similar to a throwing action.

In Figure 6 (e)(f) show predictions for car-related throwing anomalies, which are more difficult to identify for our anomaly detection method. Anomalous sections sometimes obtain a predicted anomaly score of zero. This is caused by the fact that throwing anomalies from cars are more challenging to recognize, as there could be no person-arm motions. Furthermore, the second peak with a predicted anomaly score above 0.5 in subfigure (f) corresponds to the moment that the car suddenly accelerates in the video. An explanation for this high score is that the normal training videos do not contain sufficient video material of accelerating cars, making this acceleration appear anomalous.

Finally, subfigures (g)(h) show predicted anomaly scores of two different normal videos. As can be noticed, the example in (g) is close to the ideal performance for a normal video, with a low predicted anomaly score in every segment. However, subfigure (h) indicates two false alarms, which are most likely caused by the presence of a fire truck in the footage of this normal video, because they rarely occur in any normal training video.

Overall, the experiments show that the best results for the combined UCF-Crime+Throwing dataset are achieved when training the anomaly detection model with the Adam optimizer on concatenated features, and with the proposed mean normal loss function using the mean anomaly score of normal videos. The

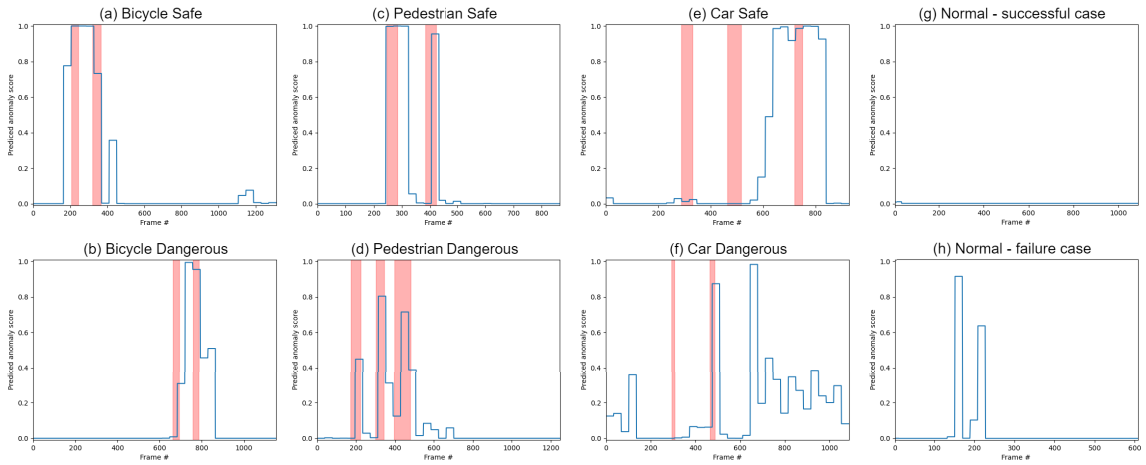

Figure 6. *Qualitative results of the anomaly detection method on testing videos from the Throwing-Action dataset. Red regions show ground-truth anomalous regions, while the blue curve depicts the predicted anomaly score as a function of the frame number. From left to right and from top to bottom, the test videos cover anomaly classes: (a) bicycle safe, (c) pedestrian safe, (e) car safe, (g) successful case normal, (b) bicycle dangerous, (d) pedestrian dangerous, (f) car dangerous. Subfigure (h) shows a failure case for the normal video class.*

	Original loss	Proposed loss
False alarm rate	0.5667	**0.4696**

Table 5. False alarm rate in percentage of normal video segments where the predicted anomaly score is more than 0.5 on the combined UCF-Crime+Throwing dataset.

UCF-Crime+Throwing dataset contains a wider range of anomalies, compared to the proposed Throwing-Action dataset. This variation explains why the concatenated features increase the performance for the combined dataset, while they do not provide an improvement for the Throwing-Action dataset. From the qualitative analysis, we conclude that our throwing anomaly detection model works well for pedestrian and bicycle-related anomalies. However, performance for car-type anomalies can be further improved by involving more data and situations appearing with cars.

False Alarm Analysis

This section separately analyzes the performance of the proposed anomaly detection method in terms of false alarm rate because it is an important indicator for the correct focus on safety.

A false alarm rate of even a few percent can render an anomaly detection algorithm useless, since by definition most of the data encountered by an anomaly detection method is normal. Therefore, the false alarm rate is an important metric for industrial applications. We compare the false alarm rates on the normal testing data of the combined UCF-Crime+Throwing dataset for models trained on concatenated features with the original loss function and the mean normal loss function, as shown in Figure 7. A false alarm is defined as a predicted anomaly score above 0.5 for a segment of a normal video. This figure shows that changing the loss function appears to positively affect the false alarm rate, reducing it significantly. With the proposed loss function, the model learns to recognize normal data faster, which results in an initial false alarm rate of zero. As the training progresses, the model learns to recognize anomalies, which sometimes introduce erroneous predictions.

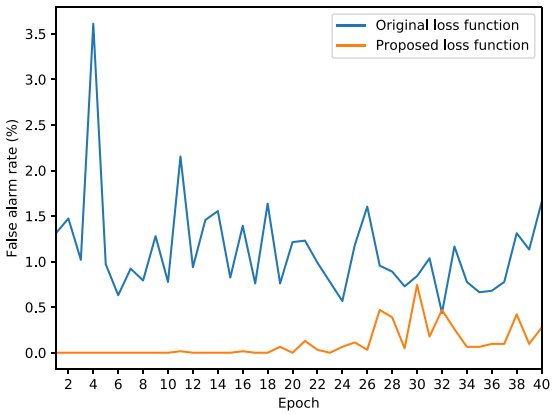

Figure 7. *False alarm rate on normal testing data for anomaly detection models trained using the original loss function and the mean normal loss function. Both models are trained on the combined UCF-Crime+Throwing dataset and false alarms are categorized as a predicted anomaly score of more than 0.5 on a normal video segment.*

Conclusion

We have introduced a novel dataset consisting of normal behaviors and throwing actions performed by car drivers, pedestrians, and bicyclists, to support the development of throwing anomaly detection algorithms. Additionally, we have compared the performance of different feature extraction networks, which are C3D, I3D-charades, I3D-kinetics and MFNet, within our anomaly detection method on the Throwing-Action dataset, the UCF-Crime dataset, and the combined dataset. Furthermore, we have found that using the Adam optimizer for training anomaly detection algorithms leads to improved results and shorter training times over the Adadelta optimizer. Finally, using the proposed mean normal loss function, we have significantly reduced the false alarm rate of the anomaly detection method, thereby improving the area under the ROC curve. Based on the experimental results,

we have successfully improved the performance on the combined dataset to an area under the ROC curve of 80.13 by means of the proposed mean normal loss function, Adam optimizer and concatenated features.

Acknowledgments

This work is supported by the European ITEA project SMART on intelligent traffic flow systems and Efficient Deep Learning (EDL) RMR project.

References

[1] J. Carreira and A. Zisserman, "Quo Vadis, Action Recognition? A New Model and the Kinetics Dataset," in *CVPR*, 2017, pp. 6299–6308.

[2] Y. Chang, Z. Tu, W. Xie, and J. Yuan, "Clustering Driven Deep Autoencoder for Video Anomaly Detection," in *Computer Vision – ECCV 2020. ECCV 2020. Lecture Notes in Computer Science*, vol. 12360 LNCS. Springer, Cham, 8 2020, pp. 329–345. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-030-58555-6_20

[3] Y. Chen, Y. Kalantidis, J. Li, S. Yan, and J. Feng, "Multi-Fiber Networks for Video Recognition," in *ECCV*, 2018, pp. 352–367.

[4] R. Csordás, L. Havasi, and T. Szirányi, "Detecting objects thrown over fence in outdoor scenes," in *Proc. Int. Conf. Comput. Vision Theory Applicat. (VISAPP)*, Berlin, 2015, pp. 593–599. [Online]. Available: http://eprints.sztaki.hu/8636/1/Csordas_593_2856278_ny.pdf

[5] Z. Dai and Z. Zheng, "A YOLOv3-Based Learning Strategy for Vehicle-Thrown-Waste Identification," in *ICIC 2021: Intelligent Computing Theories and Application*. Springer, Cham, 8 2021, pp. 305–315. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-030-84529-2_26

[6] K. Deepak, S. Chandrakala, and C. K. Mohan, "Residual spatiotemporal autoencoder for unsupervised video anomaly detection," *Signal, Image and Video Processing*, vol. 15, no. 1, pp. 215–222, 7 2020. [Online]. Available: https://link.springer.com/article/10.1007/s11760-020-01740-1

[7] S. Ioffe and C. Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift," in *Proceedings of the 32nd International Conference on Machine Learning, PMLR*, 2015, pp. 448–456.

[8] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and F. F. Li, "Large-scale video classification with convolutional neural networks," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE Computer Society, 9 2014, pp. 1725–1732. [Online]. Available: https://paperswithcode.com/paper/large-scale-video-classification-with-1

[9] E. Kosman, "Pytorch implementation of Real-World Anomaly Detection in Surveillance Videos." [Online]. Available: https://github.com/ekosman/AnomalyDetectionCVPR2018-Pytorch

[10] Z. Li, Y. Li, and Z. Gao, "Spatiotemporal Representation Learning for Video Anomaly Detection," *IEEE Access*, vol. 8, pp. 25 531–25 542, 2020.

[11] K. Liu and H. Ma, "Exploring background-bias for anomaly detection in surveillance videos," in *MM 2019 - Proceedings of the 27th ACM International Conference on Multimedia*. Association for Computing Machinery, Inc, 10 2019, pp. 1490–1499. [Online]. Available: https://doi.org/10.1145/3343031.3350998

[12] H. Lv, C. Zhou, Z. Cui, C. Xu, Y. Li, and J. Yang, "Localizing Anomalies from Weakly-Labeled Videos," *IEEE Transactions on Image Processing*, vol. 30, pp. 4505–4515, 2021.

[13] N. Nasaruddin, K. Muchtar, A. Afdhal, and A. P. J. Dwiyantoro, "Deep anomaly detection through visual attention in surveillance videos," *Journal of Big Data*, vol. 7, no. 1, pp. 1–17, 10 2020. [Online]. Available: https://journalofbigdata.springeropen.com/articles/10.1186/s40537-020-00365-y

[14] R. Nawaratne, D. Alahakoon, D. De Silva, and X. Yu, "Spatiotemporal anomaly detection using deep learning for real-time video surveillance," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 1, pp. 393–402, 1 2020.

[15] T.-N. Nguyen and J. Meunier, "Anomaly Detection in Video Sequence With Appearance-Motion Correspondence," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), 2019*, 2019, pp. 1273–1283.

[16] S. Petrocchi, G. Giorgi, and M. G. Cimino, "A Real-Time Deep Learning Approach for Real-World Video Anomaly Detection," in *ARES 2021: The 16th International Conference on Availability, Reliability and Security*. Association for Computing Machinery, 8 2021, pp. 1–9.

[17] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," *arXiv*, 4 2018. [Online]. Available: https://arxiv.org/abs/1804.02767v1

[18] E. Ribnick, S. Atev, N. Papanikolopoulos, O. Masoud, and R. Voyles, "Detection of thrown objects in indoor and outdoor scenes," in *IEEE International Conference on Intelligent Robots and Systems*, 2007, pp. 979–984.

[19] M. Sabokrou, M. Fayyaz, M. Fathy, Z. Moayed, and R. Klette, "Deep-anomaly: Fully convolutional neural network for fast anomaly detection in crowded scenes," *Computer Vision and Image Understanding*, vol. 172, pp. 88–97, 7 2018.

[20] G. A. Sigurdsson, G. Varol, X. Wang, A. Farhadi, I. Laptev, and A. Gupta, "Hollywood in Homes: Crowdsourcing Data Collection for Activity Understanding," in *ECCV*, vol. 9905 LNCS. Springer, Cham, 2016, pp. 510–526. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-319-46448-0_31

[21] P. Singh and V. Pankajakshan, "A Deep Learning Based Technique for Anomaly Detection in Surveillance Videos," in *2018 24th National Conference on Communications, NCC 2018*. Institute of Electrical and Electronics Engineers Inc., 1 2019.

[22] K. Soomro, A. Roshan Zamir, and M. Shah, "UCF101: A Dataset of 101 Human Actions Classes From Videos in The Wild," 2012. [Online]. Available: http://crcv.ucf.edu/data/UCF101.php

[23] W. Sultani, C. Chen, and M. Shah, "Real-World Anomaly Detection in Surveillance Videos," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2018, pp. 6479–6488. [Online]. Available: http://crcv.ucf.edu/projects/real-world/

[24] Y. Tian, G. Pang, Y. Chen, R. Singh, J. W. Verjans, and G. Carneiro, "Weakly-supervised Video Anomaly Detection with Robust Temporal Feature Magnitude Learning," in *International Conference for Computer Vision*, 2021.

[25] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning Spatiotemporal Features With 3D Convolutional Networks," in *ICCV*, 2015, pp. 4489–4497.

[26] W. Ullah, A. Ullah, I. U. Haq, K. Muhammad, M. Sajjad, and S. W. Baik, "CNN features with bi-directional LSTM for real-time

anomaly detection in surveillance networks," *Multimedia Tools and Applications*, vol. 80, no. 11, pp. 16 979–16 995, 8 2020. [Online]. Available: https://link.springer.com/article/10.1007/s11042-020-09406-3

[27] J. T. Zhou, J. Du, H. Zhu, X. Peng, Y. Liu, and R. S. M. Goh, "AnomalyNet: An Anomaly Detection Network for Video Surveillance," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 10, pp. 2537–2550, 10 2019.

Author Biography

Ivo Kersten obtained a B.S. degree in electrical engineering from the Eindhoven University of Technology in 2020, and a M.S. degree in electrical engineering with a specialization in artificial intelligence engineering systems from the same university in 2022. For his thesis work he conducted research on the detection of object throwing behavior in surveillance videos.

Erkut Akdag received the B.S. and M.S. degrees in electrical and electronics engineering from Middle East Technical University, Ankara, Turkey, in 2012 and 2015, respectively. He is currently working toward the Ph.D. degree at the Video Coding and Architectures Group, Eindhoven University of Technology, Eindhoven, The Netherlands. His research interests include critical vehicle detection, anomaly detection and computer vision for surveillance.

Egor Bondarev obtained his PhD degree in the Computer Science Department at TU/e, in research on performance predictions of real-time component-based systems on multiprocessor architectures. He is an Assistant Professor at the Video Coding and Architectures group, TU/e, focusing on sensor fusion, smart surveillance and 3D reconstruction. He has written and co-authored over 50 publications on real-time computer vision and image/3D processing algorithms. He is involved in large international surveillance projects like APPS and PS-CRIMSON.

Peter H.N. de With is Full Professor of the Video Coding and Architectures group in the Department of Electrical Engineering at Eindhoven University of Technology. He worked at various companies and was active as senior system architect, VP video technology, and business consultant. He is an IEEE Fellow and member of the Royal Holland Society of Academic Sciences and Humanities, has (co-)authored over 600 papers on video coding, analysis, architectures, and 3D processing and has received multiple papers awards. He has served as a program committee member of various IEEE conferences and holds some 30 patents.