UHCTD: A Comprehensive Dataset for Camera Tampering Detection

Pranav Mantini
Department of Computer Science
University of Houston
pmantini@uh.edu

Shishir K. Shah
Department of Computer Science
University of Houston
sshah@central.uh.edu

Abstract

An unauthorized or an accidental change in the view of a surveillance camera is called a tampering. Algorithms that detect tampering by analyzing the video are referred to as camera tampering detection algorithms. Most evaluations on camera tampering detection methods are presented based on individually collected datasets. One of the major challenges in the area of camera tamper detection is the absence of a public dataset with sufficient size and variations for an extensive performance evaluation. We propose a large scale synthetic dataset called University of Houston Camera Tampering Detection dataset (UHCTD) for development and testing of camera tampering detection methods. The dataset consists of a total 576 tampers with over 288 hours of video captured from two surveillance cameras. To establish an initial benchmark, we cast camera tampering detection as a classification problem. We train and evaluate three different deep architectures that have shown promise in scene classification, Alexnet, Resnet, and Densenet. Results are presented to show how the dataset can be used to train and classify images as normal, and tampered within and across cameras.

1. Introduction

An unauthorized or an accidental change in the view of a surveillance camera is called a tampering. This can be the result of a natural phenomenon or it can be induced to accomplish malicious activities. Figure 1 (a) & (b) shows an example of tampering due to a natural phenomenon, where the sunlight is reflected on to the camera lens. Figure 1 (c) & (d) shows an example of tampering induced by a human. The perpetrator blocks the view of the camera using a cardboard. Images Figure 1 (b) & (d) are compromised and may not be useful for surveillance. Another example is of natural tampering would be when cameras switch between day/night mode. Often cameras fail to auto-focus correctly after switching between day/night mode resulting in a defocussed view. Examples of unauthorized tampering would include spray painting the lens, changing the viewpoint intentionally to accomplish malicious activities (such as theft and property damage).

(a) Normal (b) Tampered (c) Normal (d) Tampered

Figure 1: Example of natural tampering due to reflection of sunlight on to the camera lens (a) & (b), Example of intentional tampering due to a covered lens (c) & (d)

Surveillance cameras have become an integral part of private and public infrastructures, and are deployed ubiquitously. Camera tampering disrupts security operations, and forensic value of captured video. Security operators generally perform manual review of cameras on a regular basis to ensure their functionality. This is a tedious and erroneous task.

Types of Camera Tampering: Most literature has classified camera tampering as covered, defocussed, and moved. Covered tampering occurs when the view of the camera is blocked. Spray painting the lens, blocking it with hand, and accumulation of dust, spider webs, and droplets are examples of covered tampering. Defocussed tampering occurs when the view of the camera is blurred. Formation of fog on lens, failures to focus, and intentionally changing the focus are examples of defocussed tampering. Moved tampering occurs when the view point of the camera has changed. This can occur as a result of strong wind, an intentionally change in the direction of the lens with malicious intent.

Camera Tampering Detection: Algorithms that detect tampering by analyzing the video are referred to as camera tampering detection algorithms. Design of a robust algorithm is a challenging problem considering that cameras are deployed to operate under varying illumination and weather conditions, they are utilized in monitoring a variety of indoor and outdoor scenes to observe and records diverse events involving humans, vehicles, and other moving elements.

Most methods cast tampering detection as a sub-problem

Copyright © 978-1-5386-9294-3/18/$31.00 ©2019 IEEE

Dataset available for download at http://qil.uh.edu/datasets

of change detection. Change detection aims at identifying abrupt changes in dynamic systems [30]. The idea assumes that certain features of images/video acquired by a camera remain consistent under normal operating conditions [18]. A large deviation in the feature indicates a tampering. A deviation in the feature could also be the result of a regular scene change. Regular changes in scene can occur due to illumination changes, weather changes, or when large objects move in front of the camera. The challenge lies in identifying or learning features that deviate sufficiently when a tampering occurs while remaining consistent under normal conditions and regular changes in scene. For a review of camera tampering methods, we direct readers to the recent work in the area [18, 25, 24].

1.1. Camera Tampering Algorithm Evaluation and Datasets

Camera tampering detection algorithms are evaluated on:

The ability to detect tampering: Tampering can occur to varying extents. For example, a covered tampering can compromise the camera view partially or completely. An algorithm will fails to detect partial tampering if it fails to capture the deviation in feature due to a partial tampering. A tampering could also be gradual or sudden. This would result in slow changes in the computed features or an abrupt change, respectively. Most algorithms leverage temporal deviation in features and rely on an abrupt change to detect tampering.

The ability to prevent false alarms: Surveillance cameras are expected to operate under varying illumination and weather conditions, for extended periods of time. In real world, tampering events occurs with a much lower probability compared to regular scene changes. It is vital to test algorithms on datasets that include regular scene changes over extended periods of time, to ascertain the algorithms performance with respect to preventing false positives.

To evaluate an algorithm, a dataset for camera tampering detection should incorporate tampers that occur to varying extents and at varying rates. To evaluate the ability to prevent false alarms, a dataset for camera tampering detection should also include regular scene changes that occur over extended periods of time. Most of the previous works have evaluated the detection accuracy of an algorithm, by inducing tampering into short videos. Akshay *et al.* [1] computed the detection accuracy on a dataset that contained 20 covered tampers in 4.5 minutes of video and 9 moved tampers in 2 minutes of video. Most methods have evaluated the detection accuracy on datasets containing anywhere between 50 to 100 short videos [22, 29, 16, 27, 12, 23, 26, 29, 15, 7, 10]. Some methods have utilized the same dataset to quantify the false positives. Mantini and Shah [17] evaluated their algorithms on the change detection dataset [4]. Akshay *et al.* [1], and Ellwart *et al.* [3] have tested for false positives on 6 hours of video while Ribnick *et al.* [19] tested for false positives on 19 hours of video. Shih *et al.* [23] has performed the most elaborate evaluation on 68 hours of video, followed by Lee *et al.* [14] on 51 hours of video. However, most of these datasets are not publicly available. One of the major challenges in the area of camera tamper detection is the absence of a public dataset with sufficient size and variation to perform a comprehensive evaluation of existing methods and serve to develop and benchmark new algorithms. Presently, there are two public datasets available for camera tampering detection.

Visual analysis of people dataset for tampering detection: Tsesmelis *et al.* [26] provided VAP's laboratory dataset consisting of 80 video sequences of 6-30 minutes at 720 X 576 image resolution, captured at 25 fps. Tampers were manually induced in the cameras and video level annotation are available for evaluation of detection accuracy.

Camera anomaly detection dataset: Wang *et al.* [29, 28] have provided a dataset with two scenarios, *normal* [29], and *traffic* [28]. The *normal* dataset consists of 75 videos including 57 videos with anomalous events and 18 video with normal events. The testing video contains 1800 to 2700 frames. Four anomalous events are simulated: camera self-defocussing, spray-painting, covering, and redirecting. The *traffic* dataset consists of 52 videos with anomalies that are similar to the ones in the *normal* dataset. The testing video contains 5400 to 10800 frames. Frame level annotations are available for the traffic dataset and video level annotations are available for the *normal* dataset.

These datasets are not sufficient to perform a comprehensive evaluation of true positive rate and false positive rate of tampering detection algorithms. Furthermore, deep learning architectures have become increasingly popular for the task of classification and recognition. These algorithms rely on a training stage that requires large amounts of data. Problems such as semantic segmentation [21], scene understanding [5] and text recognition [9] have shown the feasibility of learning from synthetic and large scale datasets. Hence, we propose a large scale synthetic dataset for camera tampering detection. The dataset consists of 576 tampers with over 288 hours (12 days, including testing and training data) of video captured from two surveillance cameras (6 days per camera). The contributions of this paper are:

1. We introduce a synthetic dataset with over 6.8 million annotated images for training and testing of tampering detection algorithms.

2. We formulate the problem of tampering detection as a classification problem, and we train and evaluate the performance of recent deep learning architectures as a benchmark.

3. We present experimental protocols to benchmark camera detection algorithms within and across cameras.

Figure 2: Row 1: Cam A & Cam B viewpoints; Row 2: Natural illumination changes in camera A; Row 3: Illumination changes due to shadow in Cam B; Row 4: Illumination changes due to weather in Cam A; Row 5: Crowded scenarios in Cam A & Cam B; Row 6: Extended periods of scene change in Cam A.

2. UHCTD: University of Houston Camera Tampering Detection Dataset

The dataset is created from two outdoor surveillance cameras, Camera A and Camera B, the viewpoints fore each are shown in Figure 2 Row 1. Camera A has a resolution of $2048X1536$, and Camera B has a resolution of $1280X960$. The videos are cropped to a 3rd of their resolution during synthesis. Camera A has a framerate of 3 frames per second (fps), and Camera B has a framerate of 10 fps. The two cameras together capture a wide variety of regular scene changes that occur in surveillance cameras. Camera tampering detection algorithms are sensitive to events that effect a large portion of pixels simultaneously. For example, illumination changes have a global effect, as do large objects passing in front of the camera. This results in a simultaneous change in a large number of pixels in the images. Such events can be inferred as tampering and often generate false alarms. Since the proposed dataset is created from surveillance cameras, they include numerous such events to allow for a comprehensive evaluation.

Specifically, the regular changes in our dataset include:
Natural illumination changes: Outdoor surveillance cameras undergo regular variations in the scene through the day due to illumination changes. The videos are captured over 6 consecutive days and include a variety of illumination changes. Figure 2 Row 2 shows examples images from Camera A, representative of the captured changes in illumination through the day. Shadows are also indicative of regular scene changes, an example of which is shown in Figure 2 Row 3 with example images captured from Camera B.
Weather related changes: Outdoor cameras undergo scene changes due to weather. Figure 2 Row 4 shows the view under rainy and overcast conditions.
Crowded scenarios: Surveillance camera can have objects that occupy a large portions of the view. Tampering detection algorithms are sensitive to such events. Our dataset consist of images with varying amounts of crowd. Figure 2 Row 5 shows example images from Cam. A and Cam. B.

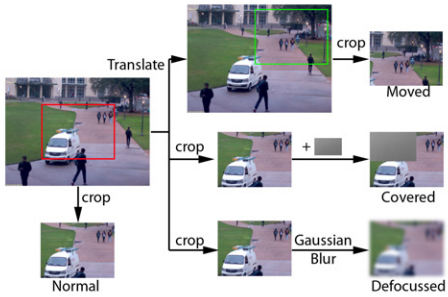
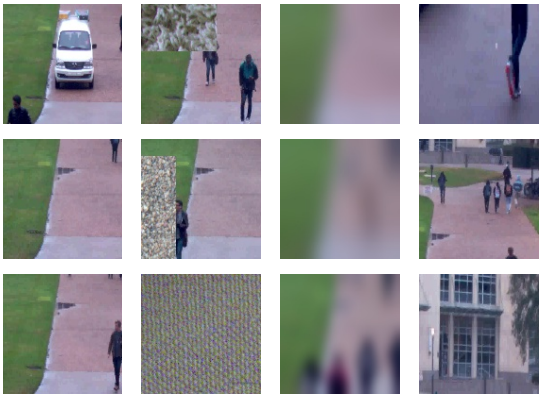


Figure 3: Tampering synthesis.

Scene change over extended period: The dataset consist of scenarios where there is a large deviation in the view occurring over an extended period of time. Figure 2 Row 6 shows an example images of such an event in Camera A, where there is a deviation caused by tents being setup for extended periods of time.



(a) left, (b) mid-left, (c) mid-right, (d) right

Figure 4: Synthetic Data. a) Original, b) Covered, c) Defocussed, and d) Moved images.

Dataset available for download at http://qil.uh.edu/datasets

The dataset provides a platform for comprehensive evaluation of tampering detection algorithms on real world surveillance videos.

2.1. Tampering Synthesis

We follow the method from Mantini and Shah [18] to simulate tampering. We synthesize tampers into the dataset using image processing techniques. Four classes of data are created (normal, covered, defocussed, and moved). We apply spatial translation, spatial smoothing, and pixel copy operations to synthesize moved, defocussed, and covered tampering respectively. Figure 3 shows the steps in synthesizing tampering in acquired images. The center region is cropped and used as a region of interest (roi) for *normal images*. Cropping the images allows us to simulate moved tampering. The roi is translated and and then cropped to simulate a *moved tampering*. The image is cropped first, and then a block of pixels are replaced by a random texture, to simulate *covered tampering*. We choose random textures from Kylberg Texture Dataset [13] to accomplish this. The image is cropped first and then smoothed using a Gaussian kernel to simulate a *defocussed tampering*. Figure 4 shows examples of normal, covered, defocussed, and moved images. We use varying range of parameters for translation, smoothing, and copying to induce tampering. Two parameters are varied to induces tampers:

Extent: The extent parameter allows us to test the capability of a system to detect varying amounts of tampering. While inducing covered tampering, extent defines the ratio of the image area that is covered. While inducing moved tampering, it defines the ratio of overlap between the original and moved image. Four different values are used to induce covered and moved tampers: $\{0.25, 0.50, 0.75, 1.00\}$. While inducing a defocussed tampering, extent represents the blurriness of the image. A 31x31 Gaussian kernel is used for inducing a defocssed tampering. Extent refers to the number of times the blurring operation is performed on the image. Four values for extent are used to induce defocussed tamper: $\{1, 11, 21, 31\}$.

Rate: The rate parameter allows us to test the ability of the system to detect tampering that occurs at different rates. Rate defines the time it takes for tampering to occur. For example, a covered tampering can occur instantaneously or over a period of five minutes. We use 6 different durations to induce tampering: $\{0.1, 0.5, 1, 5, 30, 60\}$ secs.

3. Tampering Detection as a Classification Problem

Given a reference image I_r, the test image I_t is compared against the reference image and the deviation between the two images is used to identify a tampering. Let d be some distance metric. Then a tamper is detected as: $c = \arg max_c\{p(c|d(I_t, I_r))\}$, where $c \in \{normal, covered, defocussed, moved\}$ is the class, and $p(.)$ is the posterior probability computed over the distance between the two images.

Motivated by the following observations, we cast the problem of tampering detection as a classification problem:

- It is possible to visually identify covered and defocussed images in the absence of a normal image as reference. For example, one can identify images in Figure 4 (b) & (c) as covered and defocussed, respectively, without the reference images in Figure 4 (a).
- It is not possible to define moved tamper in the absence of a normal image as reference. Nonetheless, we can identify moved tampers by introducing a fourth class that includes normal images.
- Given four classes of normal, covered, defocussed, and moved images, we can extract features from the respective images to classify them.
- Finally, recent advancements in deep neural network can be leveraged to extract discriminative feature from the image for the task of classification.

Considering this as a classification problem, a tamper is detected as: $c = \arg max_c\{p(c|I)\}$, where $c \in \{normal, covered, defocussed, moved\}$, and $p(.)$ is the posterior probability computed over the image (features). While this is a solution towards camera tampering detection, we hypothesis that one can leverage temporal information (followed by classification) as well to build robust algorithms for camera tampering detection. With this broad problem definition, we ask the following questions that are relevant to camera tampering detection in surveillance cameras.

1. How well can we learn and classify images within a single camera?
2. Can we transfer learning from one camera to another?
3. Can we learn and classify images in multiple cameras simultaneously?

4. Experimental Design

We design experiments to quantitatively answer the above questions and propose corresponding organization of our dataset.

4.1. Dataset Structure

The dataset consist of twelve days of video captured from two cameras (six days each). The data from each camera consists of a) training images, b) training video, c) testing video, and d) annotations. The structure of the dataset is shown in Table 1.

a) Training images: This part of the dataset consists of only images. Four classes of data are synthesized: normal, covered, defocussed, and moved . The tampers are induced by varying only the extent parameter. This part of the dataset does not encode any temporal information. The

Camera	Part	Length	images
Cam A	Training images	NA	51835
(fps 3)	Training video	2 X 24 Hrs	518265
(680X510)	Testing video	4 X 24 Hrs	1036512
Cam B	Training images	NA	53978
(fps 10)	Training video	2 X 24 Hrs	1727136
(426X318)	Testing video	4 X 24 Hrs	3454272

Table 1: Structure of UHCTD dataset.

Experiment	Training Data	Testing Data
1	Camera **A** Training images	Camera **A** Testing videos
2	Camera **B** Training images	Camera **A** Testing videos
3	Camera **A + B** Training images	Camera **A** Testing videos

Table 2: Experimental design.

dataset is intended for training classifiers based on image features and it consist of a uniform distribution of the four classes.

b) Training videos: This part of the dataset consist of videos. Four classes of data are simulated: normal, covered, defocussed, and moved. The tampers are induced by varying both the extent, and rate parameters. This part of the dataset encodes temporal information as well and is intended for use with methods that may require training data to learn both image features and their temporal occurrence (for example, recurrent neural networks). In the training videos, a random tamper is induced every ten minutes to ensures that the tampers are well distributed over different illuminations that occur through the day. Each tamper last between five to ten minutes.

c) Testing videos: Created similar to training video for the purpose of testing.

d) Annotations: Each frame is annotated with five value, the frame number, class, extent, rate, and progress flag. Normal images are assigned a value of 0, covered - 1, defocussed - 2, and moved - 3. According to the definition in sub-section 2.1, a value from $\{0, 25, 0.50, 0.75, 1.00\}$ is assigned to represent the extent, and a value from $\{0.1, 0.5, 1, 5, 30, 60\}$ to represent rate. Finally, progress flag indicates if a tamper has occurred or is in progression.

4.2. Experimental Design

We propose the following experiments and corresponding use of the dataset to answer the questions posed in Section 3.

Experiment 1: How well can we learn and classify images in a single camera? Given the training data from a single camera, the objective is to quantify the capability to classify normal, covered, defocussed, and moved images. We train classifiers on training images from Camera A, and test the capability of classifying the images on testing videos belonging to Camera A only. One can perform a similar experiment on data from Camera B.

Experiment 2: Can we transfer learning from one camera to another? This is a relevant question given the current deployment size of surveillance systems and the data-intensive training needs for modern classifiers based on neural networks. We train classifiers on images from Camera A, and test the capability of classifying images from Camera B. However, since moved tampering is defined based on the use of normal images, and the normal images for Camera A and Camera B are different, we perform this experiment only with covered and defocussed tampering classes.

Experiment 3: Finally, how well can we learn to classify images from multiple cameras simultaneously? This would evaluate if a single model can be trained to classify normal, covered, defocussed, and moved images taken from different cameras. We quantify this by training classifiers on data created by combining Cameras A and B, and test the capability of classifying the four classes on testing videos from Camera A. One can also conduct similar experiment and evaluate the performance on testing video from Camera B. The summary of the training and testing data is shown in Table 2.

4.3. Classification Models

Motivated by the success of deep neural network for the task of classification and recognition, we hypothesis that global features of the image can be used to classify images as normal and tampered. We explore recent architectures that have shown promise in scene classification: Alexnet [11], Resnet [6], and Densenet [8].

Alexnet is a convolutional neural network (CNN) with eight layers, the first five are convolutional and the remaining three are fully connected layers. The architecture we use is similar to the one trained on the places365 dataset [31] except that we replace the final layer that has 365 neurons with a fully connected layer with only four neurons, each to represent one of the four classes: normal, covered, defocussed, and moved. We initialize the network weights with the pre-trained weights on the places365 dataset.

Resnet is a CNN architecture that introduces shortcut connections [2, 20] that perform identity mapping between the layers [6]. We train two Resnet architectures, one with 18 layers (Resnet18), and another with 50 layers (Resnet50). Resnet18 has a two layer residual block while Resnet50 has a three layer residual block. Similar to Alexnet, we initialize the networks with pre-trained weights from network trained on the places365 dataset and replace the final fully connected layer with four neurons.

Densnet further exploits the idea of shortcuts. All the layers are connected with each other. The output from each layer is passed to the next, along with all inputs from the previ-

ous layers. We use the Densnet161 implementation, which consists of four stages of dense blocks with $6, 12, 36,$ and 24 layers, respectively. This is the largest network we train, followed by Resnet50. Similar to the previous architectures we initialize the network with pre-trained weights from network trained on the places365 dataset and replace the final fully connected layer with one that has four neurons.

We train the four networks for twenty epochs according to training data defined for experiments 1, 2, and 3.

5. Benchmark

From an application standpoint of deploying tampering detection systems on surveillance cameras, the primary objective is to identify tampering irrespective of the class (covered, defocussed, and moved). A secondary objective is to identify a detected tamper as belonging to one of the three classes. This objective allows us to understand the limitations of the system with respect to detecting each class, and build robust algorithms. To address the first objective, we quantify the performance of detecting tampering under a two-class assumption by grouping the three classes of tampering into a single class, and normal images into another class. To address the second objective, we quantify the performance of detecting each class separately.

We compute true positive rate (TPR) to quantify the systems capability to detect tampering, and false positive rate (FPR) to quantify the capability of preventing false positives. We use accuracy as a combined metric to quantify the overall performance. Lee *et al.* [15] used daily false alarm rate (DFAR) defined as false positive generated per day, as a metric to quantify false alarms. We use a similar notation, however, we notice that the methods produce a large number of false alarms. We modify the metric to compute hourly false alarm rate, which is easier to perceive, and allows for evaluation on shorter videos. Tables 3, and 4 show the performance of the four models under a two-class and four-class assumption, respectively.

Experiment 1: The results for Alexnet imply that it is possible to train and classify images as normal and tampered within a single camera. Alexnet performs best among the four models with an accuracy of 75%. The higher accuracy is attributed to the lower number of false positives it generates. This indicates that it is able to identify normal images better than other models. The order of performance for rest of the models is Resnet18, Resnset50, and Densent161. Densenet161, with a FPR of 78%, is the least capable of detecting normal images and produces the largest number of false positives. Interestingly, the idea that deeper models perform better may not translate to camera tampering detection. The large values for hFAR indicate that there is much work needed towards realizing a method that can translate to real world applications. Incorporating temporal information can assist in reducing the false alarms to a large extent. However, the underlying capability of systems to prevent false alarms must improve.

Evaluation under a four-class assumption shows that Alexnet is able to classify normal images with a higher accuracy compared to other models. Furthermore, it is capable of detecting covered tampering better, while producing lower false alarms. Resnet18 has the highest accuracy in detecting defocussed tampering. Overall Resnet based architectures produce lower false alarms while detecting defocussed tampering. The accuracy of detecting moved tampering is lower for Resnet18/50 and Densenet161. Alexnet on the other hand has a higher accuracy of detecting moved tampering, as it produces lower false positives. Overall Densnet161 tends to confuse normal images with defocussed image, and Resnet18/50 tends to confuse them with moved tampers. Alexnet, while producing lowest false positives, tends to confuse normal images with defocussed tampering.

Experiment 2: To answer if learning from one camera be used to detect tampering on a different camera, we evaluate the detection capability of covered and defocussed tampering using models trained on images from Camera A tested on images from Camera B. We ignore moved and normal class, as the moved class is defined with respect to the normal class, and the normal class is different for the two cameras. The results in Table 4 indicate that it is feasible to detect covered, and defocussed tampering without explicitly training the network on each camera. The Resnet based models outperform Alexnet and Densenet161 in this experiment. The accuracy of the Resnet models in detecting covered tampering are comparable to experiment 1. However, the accuracy for detecting defocussed tampering is degraded. Alexnet tends to produce a higher number of false positives than in experiment 1.

Experiment 3: Here the training data is the combined images from Cameras A and B. The testing data is videos from Camera A. Evaluation under a two-class assumption shows that Alexnet produces lower false positives compared to other models. While the number of false negatives increased slightly, the overall performance of Alexnet remains consistent with results from experiment 1. This may imply to a certain extent that images from multiple surveillance cameras can be used to train a single classifier for tampering detection. There is a sharp increase in false positives for the Resnet and Densenet models, and consequently an overall degradation in performance.

Evaluation under a four-class assumption shows that Alexnet's accuracy for detecting each class remains consistent with results from experiment 1. The results also indicate that the four models continue to detect covered tampering with the same accuracy as experiment 1. Resnet18/50 and Densenet161 models show a sharp degradation in the accuracy of detecting normal images. Resnet's tendency

Experiment	Arch	TP	FP	TN	FN	TPR	FPR	Acc	hFAR
1	Alexnet	229011	225984	544848	32349	0.876	**0.293**	**0.749**	**2354.0**
	Resnet18	242471	338646	432186	18889	0.927	0.439	0.653	3527.56
	Resnet50	241770	434486	336346	19590	0.925	0.563	0.560	4525.89
	Densenet161	243443	601554	169278	17917	0.931	0.780	0.399	6266.18
3	Alexnet	225548	217985	552655	35812	0.862	**0.282**	**0.754**	**2270.67**
	Resnet18	250263	514221	256419	11097	0.957	0.667	0.490	5356.46
	Resnet50	242788	697986	72654	18572	0.928	0.905	0.305	7270.68
	Densenet161	252596	738431	32209	8764	0.966	0.958	0.275	7691.98

Table 3: Comparison of the performance under a two class assumption, FP - false positives, TN - true negatives, FN - false negatives, TPR - true positive rate, FPR - false positive rate, Acc - accuracy , and hFAR - hourly false alarm rate.

		Normal			Covered			Defocussed			Moved		
Experiment	Arch	TPR	FPR	Acc	TPR	FPR	Acc	TPR	FPR	Acc	TPR	FPR	Acc
1	Alexnet	0.71	0.12	**0.75**	0.85	0.03e-2	**0.99**	1.00	0.19	0.82	0.48	0.07	**0.89**
	Resnet18	0.56	0.07	0.65	0.86	0.01	0.98	0.84	0.04	**0.95**	0.81	0.34	0.68
	Resnet50	0.44	0.07	0.56	0.87	0.02e-1	0.99	0.53	0.02	0.94	0.87	0.48	0.55
	Densenet161	0.22	0.07	0.40	0.60	0.02e-3	0.97	0.88	0.54	0.49	0.43	0.18	0.79
2	Alexnet	-	-	-	0.90	0.25	0.76	0.99	0.24	0.78	-	-	-
	Resnet18	-	-	-	0.85	0.07	0.92	0.95	0.01	**0.98**	-	-	-
	Resnet50	-	-	-	0.81	0.02	**0.96**	0.79	0.04	0.94	-	-	-
	Densenet161	-	-	-	0.58	0.03e-1	0.96	0.51	0.21	0.77	-	-	-
3	Alexnet	0.72	0.14	**0.75**	0.85	0.01e-1	**0.99**	0.99	0.16	0.85	0.47	0.09	**0.87**
	Resnet18	0.33	0.04	0.49	0.84	0.07e-1	0.98	0.88	0.02	**0.97**	0.87	0.54	0.49
	Resnet50	0.09	0.07	0.31	0.80	0.04e-2	0.98	0.62	0.01	0.96	0.94	0.76	0.29
	Densenet161	0.04	0.03	0.28	0.57	0.05e-2	0.96	0.97	0.74	0.32	0.27	0.14	0.81

Table 4: Comparison of performance under a four class assumption, TPR - true positive rate, FPR - false positive rate, and Acc - accuracy.

to confuse normal images with moved tampering has increased and so has the Densenet's tendency to confuse normal images with defocussed tampering.

6. Conclusion

This paper introduces a large scale dataset for camera tampering detection in surveillance videos. The synthetic dataset consists of over 6.8 million images. There are over 576 tamperings induced for training and testing camera tampering detection algorithms. We have posed the problem of camera tampering detection as a classification problem. We have trained and evaluated the capacity of four scene classification architectures: Alexnet, Resnet18, Resnet50, and Densenet161 to detect four classes of images: normal, covered, defocussed, and moved. The experiments show the feasibility of detecting tampering within and across multiple surveillance cameras. Alexnet is able to detect normal images with a higher accuracy over other models, and outperforms the other models with a detection accuracy of 75%. Resnet based architecture can learn and classify covered and defocussed tampering better accross multiple cameras.

ACKNOWLEDGEMENTS

This work was performed in part through the the financial assistance award, Multi-tiered Video Analytics for Abnormality Detection and Alerting to Improve Response Time for First Responder Communications and Operations (Grant No. 60NANB17D178), from U.S. Department of Commerce, National Institute of Standards and Technology.

References

[1] A. Aksay, A. Temizel, and A. E. Cetin. Camera tamper detection using wavelet analysis for video surveillance. In *2007 IEEE Conference on Advanced Video and Signal Based Surveillance*, pages 558–562, Sept 2007.

[2] C. M. Bishop et al. *Neural networks for pattern recognition*. Oxford university press, 1995.

[3] D. Ellwart, P. Szczuko, and A. Czyżewski. *Camera Sabotage Detection for Surveillance Systems*, pages 45–53. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

[4] N. Goyette, P.-M. Jodoin, F. Porikli, J. Konrad, and P. Ishwar. Changedetection. net: A new change detection benchmark dataset. In *2012 IEEE computer society conference on computer vision and pattern recognition workshops*, pages 1–8. IEEE, 2012.

[5] A. Handa, V. Patraucean, V. Badrinarayanan, S. Stent, and R. Cipolla. Understanding real world indoor scenes with synthetic data. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[6] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[7] D.-Y. Huang, C.-H. Chen, T.-Y. Chen, W.-C. Hu, and B.-C. Chen. Rapid detection of camera tampering and abnormal disturbance for video surveillance system. *Journal of Visual Communication and Image Representation*, 25(8):1865 – 1877, 2014.

[8] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

[9] M. Jaderberg, K. Simonyan, A. Vedaldi, and A. Zisserman. Synthetic data and artificial neural networks for natural scene text recognition. *arXiv preprint arXiv:1406.2227*, 2014.

[10] M. S. Javadi, Z. Kadim, H. H. Woon, M. J. Khairunnisa, and N. Samudin. Video stabilization and tampering detection for surveillance systems using homography. In *2015 International Conference on Computer, Communications, and Control Technology (I4CT)*, pages 275–279, April 2015.

[11] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[12] T. Kryjak, M. Komorkiewicz, and M. Gorgon. Fpga implementation of camera tamper detection in real-time. In *Proceedings of the 2012 Conference on Design and Architectures for Signal and Image Processing*, pages 1–8, Oct 2012.

[13] G. Kylberg. *Kylberg Texture Dataset v. 1.0.* Centre for Image Analysis, Swedish University of Agricultural Sciences and Uppsala University, 2011.

[14] G.-b. Lee, M.-j. Lee, and J. Lim. Unified camera tamper detection based on edge and object information. *Sensors*, 15(5):10315–10331, 2015.

[15] G.-b. Lee, Y.-c. Shin, J.-h. Park, and M.-j. Lee. Low-complexity camera tamper detection based on edge information, 2014.

[16] D. T. Lin and C. H. Wu. Real-time active tampering detection of surveillance camera and implementation on digital signal processor. In *2012 Eighth International Conference on Intelligent Information Hiding and Multimedia Signal Processing*, pages 383–386, July 2012.

[17] P. Mantini and S. K. Shah. A signal detection theory approach for camera tamper detection. In *Advanced Video and Signal Based Surveillance (AVSS), 2017 14th IEEE International Conference on*, pages 1–6. IEEE, 2017.

[18] P. Mantini. and S. K. Shah. Camera tampering detection using generative reference model and deep learned features. In *Proceedings of the 14th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications - Volume 5: VISAPP,*, pages 85–95. INSTICC, SciTePress, 2019.

[19] E. Ribnick, S. Atev, O. Masoud, N. Papanikolopoulos, and R. Voyles. Real-time detection of camera tampering. In *2006 IEEE International Conference on Video and Signal Based Surveillance*, pages 10–10, Nov 2006.

[20] B. D. Ripley. *Pattern recognition and neural networks*. Cambridge university press, 2007.

[21] G. Ros, L. Sellart, J. Materzynska, D. Vazquez, and A. M. Lopez. The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[22] A. Saglam and A. Temizel. Real-time adaptive camera tamper detection for video surveillance. In *2009 Sixth IEEE International Conference on Advanced Video and Signal Based Surveillance*, pages 430–435, Sept 2009.

[23] C. C. Shih, S. C. Chen, C. F. Hung, K. W. Chen, S. Y. Lin, C. W. Lin, and Y. P. Hung. Real-time camera tampering detection using two-stage scene matching. In *2013 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1–6, July 2013.

[24] A. Sidnev, M. Barinova, and S. Nosov. Efficient camera tampering detection with automatic parameter calibration. In *Advanced Video and Signal Based Surveillance (AVSS), 2018 15th IEEE International Conference on*, pages 061–066. IEEE, 2018.

[25] K. Sitara and B. Mehtre. Automated camera sabotage detection for enhancing video surveillance systems. *Multimedia Tools and Applications*, pages 1–23, 2018.

[26] T. Tsesmelis, L. Christensen, P. Fihl, and T. B. Moeslund. Tamper detection for active surveillance systems. In *2013 10th IEEE International Conference on Advanced Video and Signal Based Surveillance*, pages 57–62, Aug 2013.

[27] C.-L. Tung, P.-L. Tung, and C.-W. Kuo. Camera tamper detection using codebook model for video surveillance. In *2012 International Conference on Machine Learning and Cybernetics*, volume 5, pages 1760–1763, July 2012.

[28] Y.-K. Wang, C.-T. Fan, and J.-F. Chen. Traffic camera anomaly detection. In *Proceedings of the 2014 22Nd International Conference on Pattern Recognition*, ICPR '14, pages 4642–4647, Washington, DC, USA, 2014. IEEE Computer Society.

[29] Y. K. Wang, C. T. Fan, K. Y. Cheng, and P. S. Deng. Real-time camera anomaly detection for real-world video surveillance. In *2011 International Conference on Machine Learning and Cybernetics*, volume 4, pages 1520–1525, July 2011.

[30] A. S. Willsky. Detection of abrupt changes in dynamic systems. In *Detection of abrupt changes in signals and dynamical systems*, pages 27–49. Springer, 1985.

[31] B. Zhou, A. Lapedriza, A. Khosla, A. Oliva, and A. Torralba. Places: A 10 million image database for scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.