Camera Tampering Detection using Generative Reference Model and Deep Learned Features

Pranav Mantini and Shishir K. Shah
Department of Computer Science, University of Houston,
4800 Calhoun Road, Houston, Texas, U.S.A.
pmantini@cs.uh.edu, sshah@central.uh.edu

Keywords: Surveillance, Camera, Tampering Detection, Generative Adversarial Network, Siamese Network, Deep Learning, Surveillance Data.

Abstract: An unauthorized alteration in the viewpoint of a surveillance cameras is called tampering. This involves comparing images from the surveillance camera against a reference model. The reference model represents the features (e.g. background, edges, and interest points) of the image under normal operating conditions. The approach is to identify a tamper by analysing the distance between the features of the image from surveillance camera and from the reference model. If the distance is not within a certain threshold, the image is labeled as a tamper. Most methods have used images from the immediate past of the surveillance camera to construct the reference model. We propose to employ a generative model that learns the distribution of images from the surveillance camera under normal operating conditions, by training a generative adversarial network (GAN). The GAN is capable of sampling images from the probability density function, which are used as reference. We train a Siamese network that transforms the images into a feature space, so as to maximize the distance between the generated images and tampered images (while minimizing the distance between generated and normal images). The distance between the generated and the surveillance camera image is classified as either normal or tampered. The model is trained and tested over a synthetic dataset that is created by inducing artificial tampering (using image processing techniques). We compare the performance of the proposed model against two existing methods. Results show that the proposed model is highly capable of detecting and classifying tampering, and outperforms the existing methods with respect to accuracy and false positive rate.

1 INTRODUCTION

An unauthorized alteration in the viewpoint of a surveillance camera is called tampering. This can occur due to natural phenomena, for example, the lens can accumulate dust, it can lose focus, and its view point can shift (due to strong winds). Camera tampering can also be induced to accomplish malicious activities (like theft and property damage). Examples of such incidents include spray painting, blocking, and changing the view of the camera (with hand). Tampering detection concerns with identifying these events by analyzing the video from camera.

Detecting camera tampers safeguards security and enables forensic analysis. Today, surveillance cameras are affordable and are deployed ubiquitously. For example, universities have hundreds of cameras, international airports and casinos have thousands of cameras. They are usually deployed across the infrastructure to yield a wide coverage, and ensuring their functionality requires a continual review process. Currently, humans (security officers) perform a routine review of the surveillance cameras, but this is a tedious task, which is prone to error. Properly functioning cameras is an assumption for computer vision algorithms, such as tracking (Mantini and Shah, 2016b), re-identification (Mantini and Shah, 2015; Bedagkar-Gala and Shah, 2014), and motion prediction (Mantini and Shah, 2014; Mantini and Shah, 2016a).

Detecting tampers is a challenging problem. Surveillance cameras are expected to operate under various illumination and weather conditions, they are utilized in monitoring a variety of scenes, and they have limited access to compute power. As such, the ability to detect camera tampering requires an approach that can function successfully under varying conditions, irrespective of scene complexity while limiting false alarms and utilize minimal resources.

Camera tampering detection can be mapped to the problem of change detection, which aims at identifying abrupt changes in dynamic systems (Willsky,

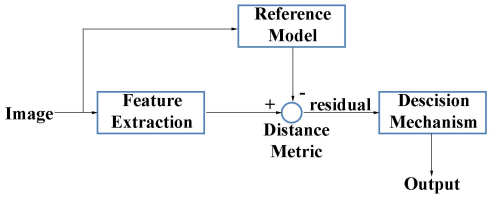


Figure 1: Residual based structure for detecting tampering.

1985). Willsky et al. (Willsky, 1985) proposed a residual based structure for detecting abrupt changes (See Figure. 1). Most methods for detecting tampering can be described using a similar structure. The framework depicts the idea that certain features of the image remain constant under normal operating conditions of the surveillance camera. A large deviation in this feature is associated with a tampering. The reference model provides the features under normal operating conditions that are compared against the feature from a test image, and a large distance between them is used to detect a tamper.

The feature extraction step isolates the features from image that the framework assumes to be constant under normal operating conditions. Some researchers have worked with the assumption that it is the background that remains unchanged, others have assumed that it is the edges, and yet a few assumed that it is the interest points (like corners). Existing methods for tampering detection can be organized as background, edge, and interest point modeling methods. Unlike the existing methods that use handcrafted features, we propose to learn features from the data for tampering detection. Recently, researches have found success in training CNNs to detect visibility loss (Ivanov and Yudin, 2019), and detecting color and intensity based abnormality (Dong et al., 2016). We propose a deep learning approach with extended ability to detect covered, defocussed, and moved tamper. While existing methods use images information from the immediate past as reference, we propose to learn the probability distribution of normal images, and sample from it to generate reference images. To compute a residual, existing methods have compared statistical measures such as mean, histogram, and entropy computed over the features of the test and reference image. We propose to transform the test and reference image to a new features space, and perform classification in this feature space to discriminate tampered from the normal images. To summarize, the contributions of this paper are:

1. Training a generative adversarial network to sample reference images from a complex probability density function.
2. Training a Siamese network to distinguish normal images from tampered images.
3. Training and evaluating performance of tampering detection on synthetic data.

2 RELATED WORK

This section describes the existing work with respect to each module in the framework.

2.1 Feature Extraction

Tampering detection can be organized as background, edge, and interest point modeling methods.

Background Modeling: Background refers to the elements of a scene that do not undergo motion. Many a method have leveraged this idea to model background as a feature for detecting tampers. Background can be modeled using efficient **frame differencing** (Collins et al., 2000; Aksay et al., 2007; Saglam and Temizel, 2009; Kryjak et al., 2012; Guler et al., 2016), **mixture of Gaussians'** (Czyzewski and Dalka, 2008; Ellwart et al., 2012), and **code books** (Tung et al., 2012; Kim et al., 2005).

Huang et al. (Huang et al., 2014) used the absolute difference between the reference and test background to compute a residual for detecting moved and covered tampers. Saglam et al. (Saglam and Temizel, 2009) modeled two backgrounds separated by a time delay to compute the residual. The first was used as a reference and the latter as the test image for detecting moved tampers. Ellwart et al. (Ellwart et al., 2012) computed the entropy of the two backgrounds and used the difference as a residual to detect covered tampering. Akshay et al. (Aksay et al., 2007) and others (Saglam and Temizel, 2009; Huang et al., 2014; Guler et al., 2016) have computed the histogram and used the concentration in the lower intensity bins of the histogram as a feature. The difference in concentrations of the reference and the test images, is used as a residual to detect covered tampering.

Edge Modeling: Edges correlate with sharp intensity changes in the image. Edges can be computed using **pixel-wise gradient** (Mantini and Shah, 2017; Harasse et al., 2004; Tsesmelis et al., 2013); **spatial filters** like Sobel (Gaibotti et al., 2015; Lee et al., 2015; Huang et al., 2014; Shih et al., 2013; Raghavan et al., 2012; Ellwart et al., 2012; Ribnick et al., 2006) and Prewitt; **frequency filters** like Gaussian high pass filter (Guler et al., 2016; Saglam and Temizel, 2009; Huang et al., 2014); and robust **edge detection** methods like canny edge detector (Lee et al., 2014; Wang et al., 2014; Lin and Wu, 2012; Ellwart et al., 2012; Wang et al., 2011; Sidnev et al., 2018). A camera

operating out-of-focus has indistinct edges. A camera that is covered or moved, results in disappearance of edges that are present in the reference image. Lin and Wu (Lin and Wu, 2012) used the intersection of edges between the test and reference image to compute a residual, and detect covered and moved tampers. Defocussing degrades edge content. Gaibotti et al. (Gaibotti et al., 2015), Mantini and Shah (Mantini and Shah, 2017) used pixel wise gradient to filter the edge content. Difference between the accumulated magnitude of the gradients is used as a residual (Gil-Jiménez et al., 2007; Wang et al., 2011; Huang et al., 2014). High frequency content in an image correspond to the sharp changes in the image. Saglam et al. (Saglam and Temizel, 2009) and Akshay et al. (Aksay et al., 2007) accumulated the co-efficient of high frequency components as a feature. Wavelet transform is applied to obtain the frequency content in the image. Huang et al. (Huang et al., 2014) and Guler et al. (Guler et al., 2016) followed a similar approach by applying discrete Fourier transform. Harasse et al. (Harasse et al., 2004) and Jiminez et al. (Gil-Jiménez et al., 2007) used the entropy of the edges as a feature for detecting covered tampering. The features described so far, quantify the magnitude of gradients/edges in the image. Ribneck et al. (Ribnick et al., 2006) used histogram of oriented gradients (HOG) as a feature. This captures the orientation of gradient as well. The sum of absolute difference between the HOGs of reference and test image is used as a residual.

A combination of background and edges can be used to extract robust features. Lee et al. (Lee et al., 2014; Lee et al., 2015) applied edge detection on the background image and used it as a feature for detecting tampering. Saglam et al. (Saglam and Temizel, 2009) used the high frequency content of the background image as a feature for detecting defocussing.

Interest Points Modeling: These methods assumes that the location of interest points in the image remain fixed under normal operating conditions. **SIFT** (Scale invariant Feature Transform) and **SURF** (Speeded Up Robust Features) are common algorithms used to identify keypoints in reference and test images. A residual is computed by comparing the two sets of interest points. Tsesmelis et al. (Tsesmelis et al., 2013) used the difference in number of interest points as a residual. Yin et al. (Yin et al., 2013) used SIFT based image descriptors as a feature, and the difference between them is used as a residual for detecting covered and moved tampers. Javadi et al. (Javadi et al., 2015) estimated the global motion by matching SIFT points between the reference and test image. The displacement is used as a residual to detect moved tampers.

2.2 Reference Model

The reference model generates the expected feature under normal operating conditions. Residual is computed by comparing this against the features of test image. The input to reference model is usually a set of images. The reference image ideally represent the camera under normal operating conditions. This data is not available. A general strategy is to assume temporal constancy. Under this assumption, frames from the immediate past are used as reference images. A common technique is to use a linear combination of the reference images to arrive at a reference value. This technique allows the system to adapt with naturally occurring illumination changes, like dusk, dawn, and etc. For example, Jimenez et al. (Gil-Jiménez et al., 2007; Ellwart et al., 2012; Aksay et al., 2007; Saglam and Temizel, 2009; Huang et al., 2014) updated the background reference image using a moving average model, and Wang et al. (Wang et al., 2011; Lin and Wu, 2012; Ellwart et al., 2012; Harasse et al., 2004) accumulated the edges over a set of frames to form reference edges.

Assuming temporal constancy has disadvantages. If images in the immediate past are tampered, then the model accumulates these features as well. The model drifts and fails to detect tampering. Adversely, the system falsely identifies normal frames as tampered. Selectivity is a common technique to avoid this, where frames identified as normal are selectively included in the model. However, performance of the system is contingent on its ability to detect tampering. In this paper, we overcome this disadvantage by proposing a generative model for estimating reference images. Until recently, it has been difficult to learn the probability density function of the images captured by a surveillance camera. Hence, generative models are not commonly practiced. However, with the recent advancement in training complex deep neural network architectures, it is possible to learn such distributions. Goodfellow et al. (Goodfellow et al., 2014) proposed a generative adversarial training scheme that can learn probability density function of the features. Generative adversarial network (GAN) is a neural network architecture that is capable of sampling features from the learned probability density function. Redford et al. (Radford et al., 2015) bridged the gap between GAN and convolutional neural networks (CNN) using deep convolutional generative adversarial network that is capable of learning a hierarchy of representations from an image dataset. This is capable of generating images with visual similarity to the training images. GANs have found applications in multiple facets. They have shown to enhance

resolution (Ledig et al., 2017), create images from text (Zhang et al., 2017), generate face images (Gauthier, 2014), and generate CT images form MRI (Nie et al., 2017). We explore its applicability pertaining to surveillance video.

2.3 Detection Mechanism

The detection mechanism analyzes the distance between features of the reference image and test image; and labels the image as either tampered or normal. It takes as input a residual value and maps it to a decision. A linear decision boundary using a thresholding scheme has been the norm (Saglam and Temizel, 2009; Gil-Jiménez et al., 2007; Wang et al., 2011; Lin and Wu, 2012; Ellwart et al., 2012; Aksay et al., 2007). Some methods have proposed multiple thresholds (Huang et al., 2014). Lee et al. (Lee et al., 2014) proposed an adaptive threshold, producing a non-linear boundary to cope with the complexity. However, a thresholding mechanism has limitations; A parameter tuning is required to choose an appropriate threshold. A non-linear decision making capability is required to cope with the complexity of surveillance camera. We propose to use a Siamese network (Bromley et al., 1994) as a detection mechanism. This allows us to create a non-linear mapping (transformation) of the input to a new feature space. The network takes as input two images and minimizes the distance between transformed features of normal image, while maximizing the distance between transformed features of the tampered, and normal images.

3 PROPOSED SOLUTION

The proposed solution is depicted in Figure 2. It consists of

- A deconvolutional neural network (generator).
- A pair of convolutional neural network (CNN) with shared weights (Siamese network).
- A fully connected neural network.

The generator takes as input a vector of random numbers. It generates an image that represents surveillance camera under normal operating condition. The image from camera at time t along with the generated image, are used as input to the pair of CNNs that share weights. This stage acts as a feature extractor for the generated and test images. The distance between transformed features are input to a fully connected neural network. The output is a posterior value estimating the probability of choosing class C given the distance between two inputs.

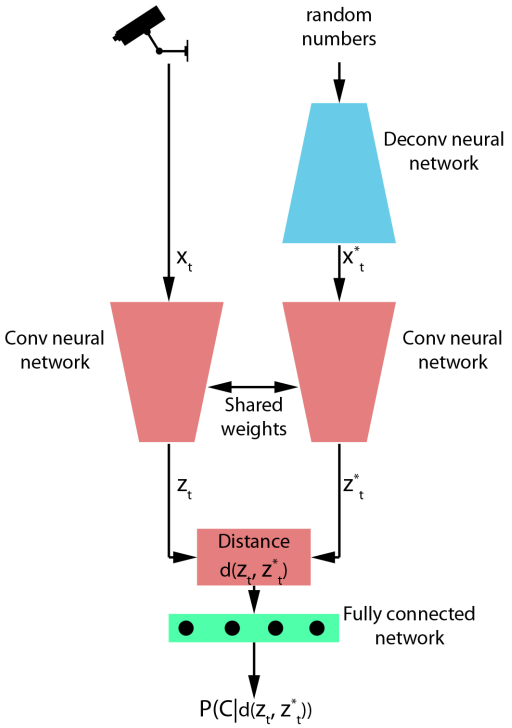


Figure 2: Proposed framework for camera tampering detection.

3.1 Generator as a Reference Model

The generator maps a latent space to a particular data distribution. It is a deconvolutional neural network composed of sequence of upsampling layers with intermittent non-linear activation layers. We train it to map the latent space to the distribution of images under normal operating condition. We train the network in an adversarial fashion. The generator is contended against another convolutional neural network, referred to as the discriminator (see Figure.3). The generator and the discriminator are involved in a zero-sum game. The generator aims at creating images that are visually similar to training example, while the discriminator aims at distinguishing the generated image from the original training image. The generator and discriminator are trained alternatively by back-propagating the error. Generative adversarial networks (GAN) have gained popularity in the recent past. We direct interested readers to Goodfellow et al. (Goodfellow et al., 2014) and Radford et al. (Radford et al., 2015) for a detailed description.

Consider G is the generator function and x^* be a sample from the generator, such that $x^* = G(y)$, where $y \sim p_y$, and p_y is a uniform distribution. Now, let us consider a discriminator (D), goal of which is to distinguish x^* from x, where x is an image from the camera. Assume that D assigns a high score to an

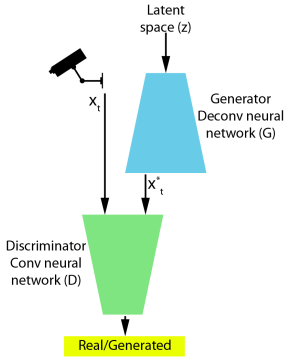


Figure 3: Generative Adversarial Network.

image, if it is the original image, and a low score if it is generated. The goal of generator is to maximize $D(G(y))$ (or minimize $1-D(G(y))$). The generator is trained to optimize the following function:

$$\min_G V_G(D,G) = \min_G(E_{y\sim p_y}[log(1-D(G(y)))]), \quad (1)$$

where V is the loss function. The discriminator maximizes the score for original images ($D(x)$) while minimizing the score for generated images, i.e. minimizing $(D(G(y))$ or maximizing $(1-D(G(y))$. The discriminator optimizes the following function

$$\max_D V_D(D,G) = \max_D(E_x[log(D(x))] + E_{y\sim p_y}[log(1-D(G(y)))]) \quad (2)$$

The combined loss function for the generator and the discriminator is,

$$\min_G \max_D V(D,G) = \min_G \max_D(E_x[log(D(x))] + E_{y\sim p_y}[log(1-D(G(y)))]) \quad (3)$$

The two networks are trained to achieve Nash equilibrium (Osborne and Rubinstein, 1994), i.e. the generator creates images that are realistic to the training data and the discriminator fails to distinguish the generated images from training images. We train the GAN on normal images from the camera. Figure 4 shows images generated by GAN ((a), (c)) and compares against the original images ((b), (d)). The two sets of images are representative of day and night. We apply log scaling on the night image.

3.2 Siamese Network as a Feature Extractor

The generator synthesizes reference images (x^*). The images from the camera (x) are compared against the synthesized images (x^*) using a distance measure that is used to detect a tamper. A linear boundary is too restricted for this task because of the complex nature

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

Figure 4: Comparison of generated images vs original images. a) generated image (daytime), b) original image (daytime), c) generated image (nighttime), and d) original image (nighttime). The night time images are log scaled.

of images from the surveillance camera. We transform x and x^* into another feature space, such that the distance between the transformed features of x and x^* is maximum, if x is tampered; and minimum, if x is normal. A linear boundary can be established in the transformed feature space to distinguish between different classes. This is modeled using a Siamese network. The network consists of two parallel convolutional network (Figure 2). The two networks share the same set of weights ($f(x,w)$), where w are the weights. The shared weights allow us to apply the same transformations on the reference (x^*) and input images (x).

$$z = f(x,w) \; and \; z^* = f(x^*,w) \quad (4)$$

We compare the transformed features using a distance measure ($d(z,z*)$). The distance vector is used as input to a fully connected layer. This estimates the posterior values of the image belonging to each class ($p(C|d(z,z*))$). We follow a maximum posterior approach to choose class o.

$$o = \arg_i \max p(c_i|d(z,z^*)) \quad (5)$$

The network is trained to minimize the categorical cross entropy loss.

$$\min_S V_S = \min_S \sum_{i=1}^{N} \sum_{c=1}^{k} 1_{o \in c_i} log(p(o \in c_i)) \quad (6)$$

Where $1_{o \in c_i}$ is the indicator function.

4 IMPLEMENTATION

We discuss the system architecture, training procedure, and the dataset in this section.

4.1 System Architecture

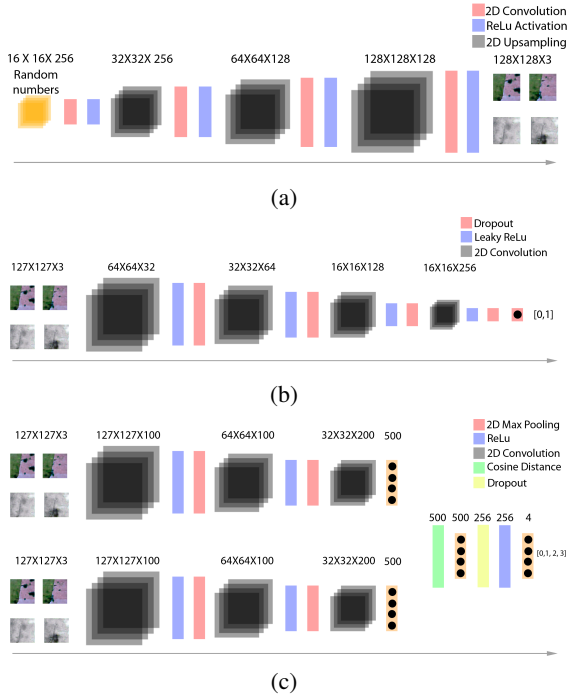


Figure 5: System architecture. a) Generator, b) Discriminator, and c) Siamese.

Figure 5(a) shows the system architecture of the generator. The input is a matrix of random numbers with size $16X16X256$, the numbers are passed through three 2D upsampling layers. Each upsampling layer is followed by a 2D convolution layer. We perform batch normalization and apply activation using rectified linear units (ReLu). The output is a matrix of size $127X127X3$. We choose this size to match the size of the images in the dataset. Figure 5(b) shows the system architecture of the discriminator used in the adversarial training of the generator. It consists of four 2D convolution layers. We perform batch normalization and use leaky ReLu for activation after each convolution layer. We allow a 25% dropout after each activation to avoid over-fitting. Figure 5(c) shows the architecture of the Siamese network. The base CNN consists of two convolution layers; each followed by ReLu activation and 2D max pooling layers. A pair of these CNNs are initialized; one takes the input from the generator, and the other takes the input from the camera. We compute the cosine distance between the output vector from the two CNNs. The distance vector is given as input to a fully connected layer followed by a dropout and ReLu activation layer. Finally passed through another fully connected layer, the output of which, is mapped to posterior values of the four classes using softmax activation. The four classes represent normal, covered, defocussed, and moved status of the camera.

4.2 Training Approach

The generative adversarial network and the Siamese network are trained separately. The dataset required for training the two networks is different. The GAN requires a large number of images, representing the normal operating conditions of the camera. The Siamese network, requires a combination of normal, covered, defocussed, and moved images. Large scale datasets representing camera tampering in surveillance camera are not available.

Training the GAN: The GAN is trained using images from a normally functioning surveillance camera. The GAN fails to learn representation of images that are contrasting. For example, we failed to train a GAN to sample images that represent both day and night (Fig 6 (a, b) respectively). The training data is segmented into multiple clusters based on their color features using K-means. (Fig 6 (c, d)) shows the representation of the centers of the clusters for $K = 2$. While testing, the suitable GAN is selected based on the image's distance to the cluster.

Fig 6 (e, f) shows sample images generated by the two GANs. Furthermore, the GAN fail to learn the distribution for night images because of their low dynamic range. We apply log scaling, on the images in the training and testing phases. Figure 6 (b) show the night image after log scaling.

Training the Siamese Network: We synthesize the data required for training the Siamese network, using image processing techniques. Four classes of data are required for training the Siamese network. We apply spatial translation, spatial smoothing, and pixel copy operations to synthesize moved, defocussed, and covered tampers; respectively. Figure 7 shows the steps in synthesizing tampering in surveillance images. The

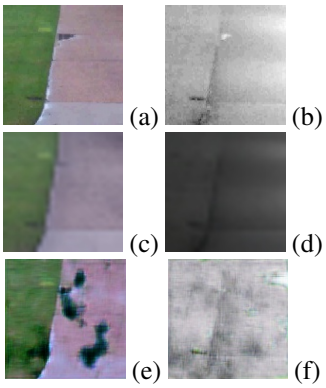


Figure 6: Surveillance camera view a) Day, b) Night; Kmeans center: c) K = 0, d) K = 1; Generated Images e) K = 0, and f) K = 1.

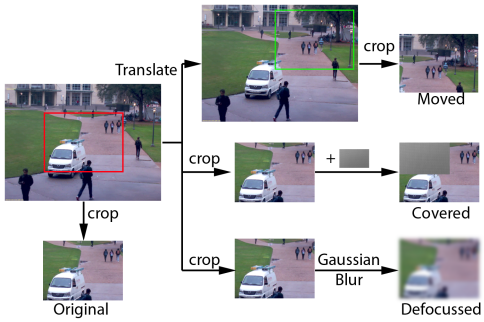
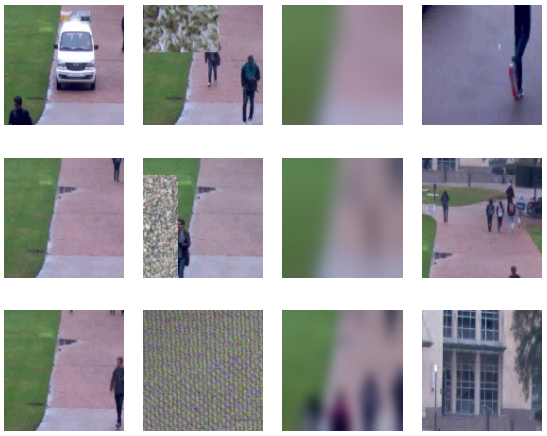


Figure 7: Synthetic Data: Method to synthesize original, covered, defocussed, and moved images.

center region is cropped and used as a region of interest (roi) for normal images; cropping the images allows us to simulate moved tampers. The roi is translated and and then cropped to simulate a moved tamper. The image is cropped first, and then a block of pixels are replaced by a random texture, to simulate covered tampers. We choose random textures from Kylberg Texture Dataset (Kylberg, 2011) to accomplish this. The image is cropped first, and then smoothed using a Gaussian kernel to simulate a defocussed tamper. We use various range of parameters for translation, smoothing, and copying to induce various extents of tampering. Figure 8 shows examples of original, covered, defocussed, and moved images.

We use normal images captured to train the GAN, and synthetic data containing a uniform distribution of four classes to train the Siamese network. The data are captured from a surveillance camera operating at 3 fps. Twenty four hours of data are used to train the GANs and the Siamese network. We split the data into two cluster for day and night to train individual GANs and the Siamese network. The GANs and Siamese network are trained over 5 and 10 epochs, respectively.



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

Figure 8: Synthetic Data. a) Original, b) Covered, c) Defocussed, and d) Moved images.

4.3 Evaluations

We evaluate performance of the system over a testing dataset that is representative of video from a surveillance camera. Testing dataset consists of video from surveillance camera, captured over a period of 24 hours at 3 fps. The images are originally captured at $1080p$ and down sampled to $127X127$. The dataset consists of over $250K$ images of which a quarter ($65K$) are tampered; the tampers are distributed uniformly (approximately $21K$ each).

The testing data are synthetic in nature. A random tamper is induced every ten minutes over a period of 24 hours; this ensures that the testing data are well distributed over different illumination changes occurring through the day. Each tamper last between five to ten minutes. Two parameters are varied to induces tampers: extent, and rate. The extent parameter allows us to test the ability of the system to detect varying amounts of tampering. In covered tampering, extent defines the ratio of the image area that is covered. In defocussed images, it defines the blurriness of the image. In moved images, it defines the ratio of overlap between the original and moved image. The rate parameter, allows us to test the ability of the system to detect tampering that occurs at different rate. Rate defines the time it takes for tampering to occur. For example, a covered tampering that covers the entire image can occur instantaneously or over a period of five minutes.

We compare the performance of the proposed system against two existing approaches, which are (Mantini and Shah, 2017) and (Lee et al., 2014). (Mantini and Shah, 2017), and (Lee et al., 2014) are unified tampering detection systems. They have the capability to detect tampering, but lack the capacity

Table 1: Performance comparison against (Lee et al., 2014) and (Mantini and Shah, 2017), where, TP - true positives, FP - false positives, TN - true negatives, FN - false negatives, TPR - true positive rate, FPR, false positive rate, and Acc - accuracy.

	Method	TP	FP	TN	FN	TPR	FPR	Acc
Covered	(Lee et al., 2014)	20213	189383	4405	1567	0.928	0.977	0.114
	(Mantini and Shah, 2017)	14704	7720	186068	7076	0.675	0.039	0.931
	(Proposed)	21599	22286	171474	181	**0.991**	0.115	0.895
	(Proposed$_2$)	27889	5233	166712	1980	0.933	**0.030**	**0.964**
Defocus	(Lee et al., 2014)	21502	189383	4405	278	0.987	0.977	0.120
	(Mantini and Shah, 2017)	18929	7720	186068	2851	0.869	0.039	0.950
	(Proposed)	21508	22286	171474	272	**0.987**	0.115	0.895
	(Proposed$_2$)	26892	5233	166712	2449	0.916	**0.030**	**0.961**
Moved	(Lee et al., 2014)	20306	189383	4405	1474	0.932	0.977	0.114
	(Mantini and Shah, 2017)	2966	7720	186068	18814	0.136	0.039	0.876
	(Proposed)	21757	22286	171474	23	**0.998**	0.115	0.896
	(Proposed$_2$)	25567	5233	166712	2373	0.915	**0.030**	**0.961**
Overall	(Lee et al., 2014)	62021	189383	4405	3319	0.949	0.977	0.256
	(Mantini and Shah, 2017)	36599	7720	186068	28741	0.560	0.039	0.859
	(Proposed)	64864	22286	171474	476	**0.992**	0.115	0.912
	(Proposed$_2$)	80348	5233	166712	6802	0.921	**0.030**	**0.953**

to classify the tampering as covered, defocussed, and moved. We perform comparison under a two class assumption of normal, and tampered.

Most methods so far have relied on image's appearance features to detect tampering, and temporal analysis to suppress false positives. (Mantini and Shah, 2017) used gradient magnitude as feature to detect tampering, and a Kalman filter to suppress false positives. (Lee et al., 2014) uses edges as features to detect tampering, and a moving average mechanism to suppress spurious false alarms. The proposed system uses image's features to detect tampering, and does not conduct any temporal analysis. While the core objective of this paper is in understanding the appearance features of an image that can be leveraged for detecting tampering, in order to compare it with existing methods, we supplement the proposed system with a simple temporal analysis mechanism to suppress spurious false alarms, and demonstrate the results for the two class scenario ($c \in [0,1]$). Proposed$_2$, detects a tamper at time t, by taking the mode of the class predictions from the previous $t-n$ instances.

$$c_t = mode(c_{t-1}, c_{t-2}, ..., c_{t-n}) \tag{7}$$

Proposed$_2$ uses $n = 3$.

We quantify the performance of detecting the three classes of tampering by dividing the dataset into three sets, each containing of normal images and only one variety of tamper (covered, defocussed, and moved). Table 1 shows the performance of the proposed system against (Lee et al., 2014) and (Mantini and Shah, 2017).

Results show Proposed$_2$ to perform best with an ovarall accuracy of 95%, followed by the proposed method with 91%, (Mantini and Shah, 2017) with 85%, and (Lee et al., 2014) with 25%. It should be noted that the (Lee et al., 2014) is unable to cope with the complexity of the scene and eventually labels all images as tampered. This is evident from the 97% false positive rate. Mantini and Shah's (Mantini and Shah, 2017) method is capable of detecting defocussed images better than other tampers. Their method models signal activity, an accumulated measure of gradient magnitude, to detect tampering. The edges degrade under a defocussed tamper but not necessarily during a covered or moved tamper. Their system performs poorly while detecting moved tampers. (Mantini and Shah, 2017) shows a higher accuracy for covered and defocussed images, and produced less false positives as well, compared to the proposed system. This can be attributed to the systems ability to suppress false alarms, rather than its ability to detect tampering. This can be demonstrated from the fact that Proposed$_2$ method outperforms (Mantini and Shah, 2017) with respect to accuracy and false positive rate because it is supplemented to perform temporal analysis. The proposed method is highly capable of detecting tampering, it detected 99%, 98% and 99% of covered, defocussed, and moved tampers respectively, while Proposed$_2$ detected 93%, 91% and

Table 2: Confusion matrix.

	Normal	Covered	Defocus	Moved
Normal	171474	6872	8119	7295
Covered	181	21383	185	31
Defocus	272	85	21011	412
Moved	23	1529	26	20202

91%, and (Mantini and Shah, 2017) detected 67%, 86% and 13%. Temporal analysis lowers the false positives, but effects the systems capability to detect tampering.

The proposed system classifies the image as one of the four classes: normal, covered, defocussed, and moved. Table 2 shows the confusion matrix, and there is a noticeable confusion amongst the three tampering classes and normal images. These correspond to the false alarms. The false negatives are minimal. Five percent of moved images have been classified as covered, and two percent of defocussed images are classified as moved. Overall, the system is highly capable of detecting and classifying tampers.

4.4 Disadvantages

The proposed system requires a large dataset for training, compared to (Mantini and Shah, 2017; Lee et al., 2014). Most methods use selectivity to update the moving average and the reference model. The proposed method does not formally introduce an online mechanism to update the trained model. So, the performance of the system under extreme weather conditions is unpredictable. Furthermore, this renders the system scene dependent. While the existing methods are also scene dependent, their training phases are quick involving only a few hundred frames. This is not viable with the proposed system. We wish to explore the performance of the system using scene independent features in future, and validate if the learning can transfer to various scenes.

5 CONCLUSION

This paper presents a novel deep learning approach for detecting tampering in surveillance cameras. In the process we have trained a generative adversarial network that is capable of sampling images, which represent the normal operating conditions of the camera. We trained a Siamese network that transforms image into a new feature space so as to maximize the distance between normal and tampered images, while minimizing the distance amongst normal images. This paper shows a method to induce artificial tampering in images to synthesize tampering in surveillance videos. We have compared the performance of the proposed system against (Mantini and Shah, 2017), and (Lee et al., 2014). Results show that the model is highly capable of detecting and classifying tampering, and outperforms two existing methods with respect to accuracy and false positive rate.

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

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

Bedagkar-Gala, A. and Shah, S. K. (2014). A survey of approaches and trends in person re-identification. *Image and Vision Computing*, 32(4):270 – 286.

Bromley, J., Guyon, I., LeCun, Y., Säckinger, E., and Shah, R. (1994). Signature verification using a" siamese" time delay neural network. In *Advances in neural information processing systems*, pages 737–744.

Collins, R. T., Lipton, A. J., Kanade, T., Fujiyoshi, H., Duggins, D., Tsin, Y., Tolliver, D., Enomoto, N., Hasegawa, O., Burt, P., et al. (2000). A system for video surveillance and monitoring. Technical report, Technical Report CMU-RI-TR-00-12, Robotics Institute, Carnegie Mellon University.

Czyzewski, A. and Dalka, P. (2008). Moving object detection and tracking for the purpose of multimodal surveillance system in urban areas. In *New Directions in Intelligent Interactive Multimedia*, pages 75–84. Springer.

Dong, L., Zhang, Y., Wen, C., and Wu, H. (2016). Camera anomaly detection based on morphological analysis and deep learning. In *2016 IEEE International Conference on Digital Signal Processing (DSP)*, pages 266–270.

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

Gaibotti, A., Marchisio, C., Sentinelli, A., and Boracchi, G. (2015). *Tampering Detection in Low-Power Smart*

Cameras, pages 243–252. Springer International Publishing, Cham.

Gauthier, J. (2014). Conditional generative adversarial nets for convolutional face generation. *Class Project for Stanford CS231N: Convolutional Neural Networks for Visual Recognition, Winter semester*, 2014(5):2.

Gil-Jiménez, P., López-Sastre, R., Siegmann, P., Acevedo-Rodríguez, J., and Maldonado-Bascón, S. (2007). *Automatic Control of Video Surveillance Camera Sabotage*, pages 222–231. Springer Berlin Heidelberg, Berlin, Heidelberg.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680.

Guler, P., Emeksiz, D., Temizel, A., Teke, M., and Temizel, T. T. (2016). Real-time multi-camera video analytics system on gpu. *Journal of Real-Time Image Processing*, 11(3):457–472.

Harasse, S., Bonnaud, L., Caplier, A., and Desvignes, M. (2004). Automated camera dysfunctions detection. In *6th IEEE Southwest Symposium on Image Analysis and Interpretation, 2004.*, pages 36–40.

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

Ivanov, A. and Yudin, D. (2019). Visibility loss detection for video camera using deep convolutional neural networks: Volume 1. pages 434–443.

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

Kim, K., Chalidabhongse, T. H., Harwood, D., and Davis, L. (2005). Real-time foreground–background segmentation using codebook model. *Real-time imaging*, 11(3):172–185.

Kryjak, T., Komorkiewicz, M., and Gorgon, M. (2012). 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.

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

Ledig, C., Theis, L., Huszár, F., Caballero, J., Cunningham, A., Acosta, A., Aitken, A. P., Tejani, A., Totz, J., Wang, Z., et al. (2017). Photo-realistic single image super-resolution using a generative adversarial network. In *CVPR*, volume 2, page 4.

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

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

Lin, D. T. and Wu, C. H. (2012). 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.

Mantini, P. and Shah, S. K. (2014). Human trajectory forecasting in indoor environments using geometric context. In *Proceedings of the 2014 Indian Conference on Computer Vision Graphics and Image Processing*, page 64. ACM.

Mantini, P. and Shah, S. K. (2015). Person re-identification using geometry constrained human trajectory modeling. In *Technologies for Homeland Security (HST), 2015 IEEE International Symposium on*, pages 1–6. IEEE.

Mantini, P. and Shah, S. K. (2016a). Camera placement optimization conditioned on human behavior and 3d geometry. In *VISIGRAPP (3: VISAPP)*, pages 227–237.

Mantini, P. and Shah, S. K. (2016b). Multiple people tracking using contextual trajectory forecasting. In *Technologies for Homeland Security (HST), 2016 IEEE Symposium on*, pages 1–6. IEEE.

Mantini, P. and Shah, S. K. (2017). 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.

Nie, D., Trullo, R., Lian, J., Petitjean, C., Ruan, S., Wang, Q., and Shen, D. (2017). Medical image synthesis with context-aware generative adversarial networks. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 417–425. Springer.

Osborne, M. J. and Rubinstein, A. (1994). *A course in game theory*. MIT press.

Radford, A., Metz, L., and Chintala, S. (2015). Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*.

Raghavan, A., Price, R., and Liu, J. (2012). Detection of scene obstructions and persistent view changes in transportation camera systems. In *2012 15th International IEEE Conference on Intelligent Transportation Systems*, pages 957–962.

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

Saglam, A. and Temizel, A. (2009). 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.

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

Sidnev, A., Barinova, M., and Nosov, S. (2018). 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.

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

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

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

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

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

Yin, H., Jiao, X., Luo, X., and Yi, C. (2013). Sift-based camera tamper detection for video surveillance. In *2013 25th Chinese Control and Decision Conference (CCDC)*, pages 665–668.

Zhang, H., Xu, T., Li, H., Zhang, S., Huang, X., Wang, X., and Metaxas, D. (2017). Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. *arXiv preprint*.