mathematics

MDPI

Article

DEEP-STA: Deep Learning-Based Detection and Localization of Various Types of Inter-Frame Video Tampering Using Spatiotemporal Analysis

Naheed Akhtar [1], Muhammad Hussain [2] and Zulfiqar Habib [1,*]

[1] Department of Computer Science, COMSATS University Islamabad, Lahore Campus, Islamabad 45550, Pakistan; sp19-pcs-009@cuilahore.edu.pk
[2] Department of Computer Science, King Saud University, Riyadh 11543, Saudi Arabia; mhussain@ksu.edu.sa
* Correspondence: drzhabib@cuilahore.edu.pk

Abstract: Inter-frame tampering in surveillance videos undermines the integrity of video evidence, potentially influencing law enforcement investigations and court decisions. This type of tampering is the most common tampering method, often imperceptible to the human eye. Until now, various algorithms have been proposed to identify such tampering, based on handcrafted features. Automatic detection, localization, and determine the tampering type, while maintaining accuracy and processing speed, is still a challenge. We propose a novel method for detecting inter-frame tampering by exploiting a 2D convolution neural network (2D-CNN) of spatiotemporal information and fusion for deep automatic feature extraction, employing an autoencoder to significantly reduce the computational overhead by reducing the dimensionality of the feature's space; analyzing long-range dependencies within video frames using long short-term memory (LSTM) and gated recurrent units (GRU), which helps to detect tampering traces; and finally, adding a fully connected layer (FC), with softmax activation for classification. The structural similarity index measure (SSIM) is utilized to localize tampering. We perform extensive experiments on datasets, comprised of challenging videos with different complexity levels. The results demonstrate that the proposed method can identify and pinpoint tampering regions with more than 90% accuracy, irrespective of video frame rates, video formats, number of tampering frames, and the compression quality factor.

Keywords: inter-frame tampering; spatiotemporal; 2D-CNN; autoencoder; dimensionality reduction; LSTM; GRU; frame insertion/deletion detection

MSC: 68T07

Citation: Akhtar, N.; Hussain, M.; Habib, Z. DEEP-STA: Deep Learning-Based Detection and Localization of Various Types of Inter-Frame Video Tampering Using Spatiotemporal Analysis. *Mathematics* **2024**, *12*, 1778. https://doi.org/10.3390/math12121778

Academic Editor: Ali Al Bataineh

Received: 25 April 2024
Revised: 30 May 2024
Accepted: 3 June 2024
Published: 7 June 2024

1. Introduction

Currently, surveillance cameras are used everywhere, and their recorded videos are often used as electronic evidence to strengthen certain claims in forensic statements or criminal investigations, enhancing people's understanding of the described incident. However, this evidence remains valid only if the content represented in the digital video is genuine. It is now easy to edit video content due to the accessibility of video editing software, such as Final Cut Pro, Adobe Premiere Pro 22.2, iMovie 3.0, Movie Maker 16.4.3528.331, and Adobe After Effects 24.0.1, without leaving any perceptible traces. While such software is primarily intended to enhance the visual quality of video content, it opens the door for potential misuse, enabling individuals to create malicious content. This poses a significant risk to sensitive fields such as science, law, or medical fields, which depend heavily on authentic digital content. The evolution of video editing tools introduces new challenges for digital forensics. Digital video serves as a powerful medium for delivering educational content and aiding in skill acquisition. Its authenticity is crucial, particularly when this recorded footage serves as primary evidence in legal investigations, criminal

trials, and judicial proceedings. With simple video editing tools, an attacker can easily delete events, alter the chronological order of events, or duplicate frames to create a fake version that seems genuine. These tampered videos can be used to mislead the police and the courts' decisions. The research community has proposed novel techniques to address these emerging challenges. Since these tamperings are often imperceptible to the human eye, verifying the authenticity and integrity of multimedia data such as graphics, audio, and videos appearing on different social networking sites has become a major challenge for researchers, scientists, and investigative agencies. The types of video tampering detection techniques include (1) active techniques and (2) passive techniques (also known as blind methods). Active techniques rely on known traces such as digital signatures or watermarks embedded into the content during the acquisition phase when the video is being recorded or later during the transmission of data. Any change in this embedded information indicates tampering. This technique may fail when the alteration is performed before inserting a digital signature or watermark [1–3]. Passive techniques are further categorized into inter-frame and intra-frame tampering detection techniques [4–7]. Both of these methods involve manipulating video content but target different domains. Intra-frame tampering (spatial tampering) involves manipulating the individual frames of a video, which can be detected by using image forensics algorithms. Common types of intra-frame forgeries include copy-move and region splicing [7]. Inter-frame forgeries (temporal tampering), on the other hand, involve manipulating the video content between frames in a video sequence. In our research, we focus on inter-frame tampering, a technique commonly applied to surveillance videos because it is easy to execute and almost imperceptible. Frame duplication, deletion, and insertion are the most commonly used semantic-focused operations in surveillance videos, as illustrated in Figure 1. In frame insertion, frames from a different video are introduced to add a fake event. Some sequence of frames is deleted in frame deletion tampering to hide an event. Frame duplication contains the repetition of an event. Such tampered videos can mislead investigators, especially in criminal investigations [8,9]. Therefore, there is an urgent need to verify that the content is genuine, and that it accurately represents reality. This problem requires the development of a robust tampering detection system to combat malicious video tampering.

Various video tampering detection techniques have been proposed in the literature to detect inter-frame tampering; these techniques are based on extracting manual features, such as statistical features [10–12], pixel and texture characteristics [13–15], motion residual, and optical flow [16–18], and a few are based on deep learning [8,19–21]. The manual features are sensitive to post-processing operations like blurring, brightness, noise, and compression. Additionally, most existing approaches examine traces to detect only one type of inter-frame tampering such as frame cloning [5], frame deletion [16,22], frame shuffling, frame insertion, and frame duplication [15,23,24] and thus, cannot simultaneously detect all kinds of inter-frame tampering, along with their types. These limitations hinder their performance in real-world applications.

Despite various proposed solutions for detecting inter-frame tampering, four major challenges remain. First, there is limited applicability; many video tampering detection techniques are restricted by factors like the number of tampered frames, frame rate, and video format, which limits their practical use [25,26]. For example, the deep learning-based method proposed in Ref. [21] can only detect inter-frame tampering if the tampered frames exist in multiples of 10, failing if there are fewer than 25 tampered frames. Similarly, the method proposed by Bakas and Naskar in [27] cannot detect frame duplication of more than 20 frames.

Second, there is poor generalization; in order to evaluate video tampering detection algorithms, benchmark datasets are crucial [28]. Many researchers have developed their personal datasets [16,17,22] to perform experiments to detect inter-frame tampering, but these datasets are not made available to the public and the research community and are often small in size. Due to the unavailability of benchmark datasets, cross-validation has

not been performed; thus, the generalization capability of existing methods cannot be ensured [14,25].

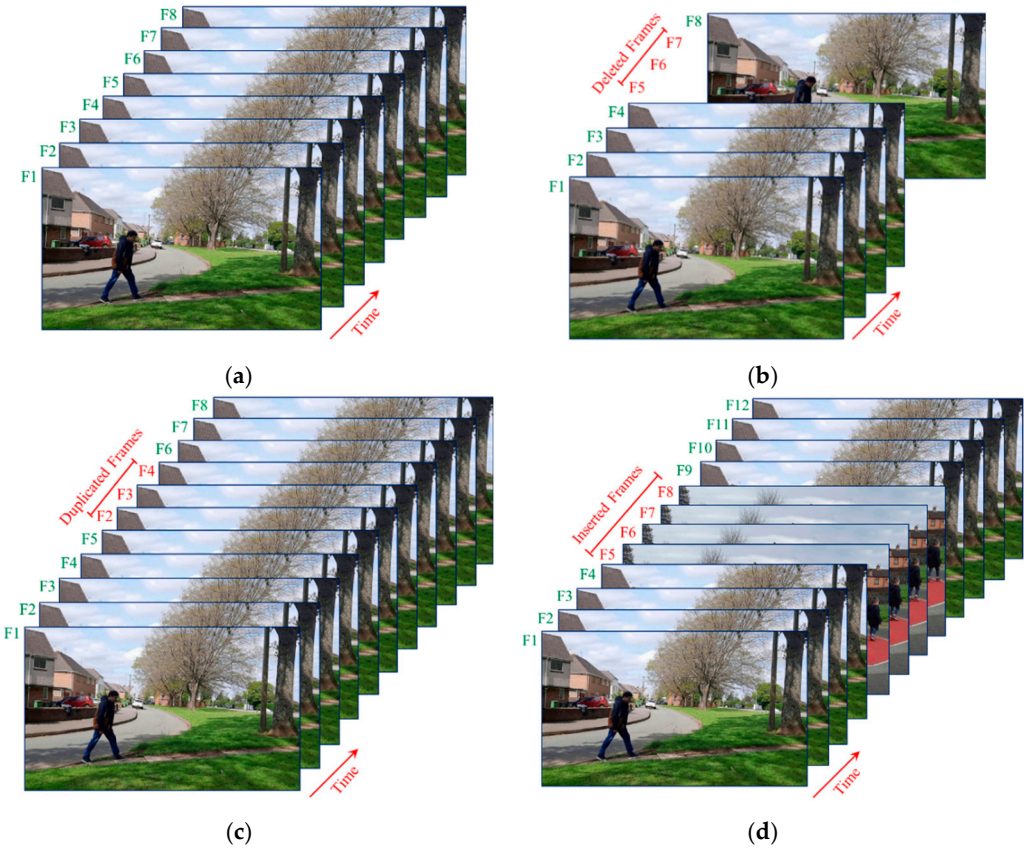


Figure 1. Inter-frame tampering: (**a**) original frames; (**b**) frame deletion; (**c**) frame duplication; (**d**) frame insertion.

Third, there is the challenge of forgery localization; some methods detect deletion only at specific positions within a video shot, such as the method in Ref. [29], which detects frame deletion forgery only at the center of a 16-frame video shot; however, the frames are not always deleted in the middle portion of a video.

Fourth, there is high computational complexity; many state-of-the-art approaches are computationally intensive because they rely on pixel-based [30,31], spatial and/or temporal correlation-based [32–34], or high dimensional feature-based methods [35–37], making it time-consuming to analyze high resolution or lengthy videos. Due to these challenges, there is a dire need for a video tampering detection system that meets these basic requirements: high accuracy, strong applicability, and high generalization capability, with good robustness. In order to address these drawbacks, we propose a novel forensic system which is capable of detecting and localizing multiple types of inter-frame tampering using spatiotemporal analysis, based on deep learning. It selects the robust features using a state-of-the-art 2D convolution neural network (2D-CNN) and the fusion of spatiotemporal information. To deal with high dimensional features and computational complexity, an autoencoder is utilized to reduce the dimensionality of feature space. Moreover, special types of recurrent neural networks (RNN), like long short-term memory (LSTM) and gated recurrent units (GRU), are used to handle long-term dependencies and input sequences of variable length, performing well with time series data. Finally, a fully connected (FC) layer, with softmax activation, is added to LSTM/GRU, yielding posterior probabilities of the classes. The major highlights of the proposed work are presented as follows:

- We proposed a robust video tampering detection method, which first extracts discriminative features using a CNN model and then takes into account the interde-

pendencies of frames to detect tampering traces in videos due to frame deletion and insertion. It detects deletion and insertion simultaneously, unlike the state-of-the-art methods [8,29,38], which detect only one type of video tampering. Moreover, the proposed technique does not impose any constraint on the minimum number of inserted/deleted frames in a video to make the tampering detectable; it can detect the insertion and deletion of as few as ten frames, along with the type of tampering. On the contrary, the method in Ref. [21] detects tampering if tampered frames exist in multiples of 10 and cannot detect tampering of less than 25 frames.

- For the proposed method, we introduced an efficient feature extraction method that first uses spatiotemporal average pooling (STP) of overlapping video clips and then employs a pre-trained CNN model such as VGG-16 as a feature extractor. This approach harnesses the hierarchical structure of the CNN model to extract rich and deep features. Our method demonstrates superior performance compared to the state-of-the-art techniques.
- The dimension of the features is very high, which causes computational difficulties. To overcome this issue, we propose to use an autoencoder to reduce the dimensionality of feature space. This significantly lessens the computational overhead of the proposed method by reducing the dimensionality of the feature space.
- We analyze the long-range dependencies among the video frames using LSTM/GRU to detect tampering traces; this leads to high accuracy in detecting tampering in videos, irrespective of their frame rates, video formats, number of tampering frames, and compression quality factor.
- The rest of the paper is outlined as follows: Section 2 strengthens this research with a review, showing the gaps in the existing research in this field. Sections 3–5 represent the proposed method, dataset description, and experiments, along with the results, respectively. Finally, in Section 6, the conclusions, along with future directions, are presented.

2. Literature Review

In the realm of multimedia forensics, the challenge of detecting video tampering remains at a nascent stage. There is a lack of robust techniques that can detect and localize video inter-frame tampering [5,39]. Several significant approaches have been introduced in digital multimedia forensics, which can be broadly divided according to their feature extraction methods: handcrafted-based methods and deep learning-based methods.

In handcrafted-based methods, many researchers have proposed numerous techniques that use both temporal and spatial correlations of overlapped video clips, and the similarity was determined to detect frame duplication tampering [14,40,41]. Lin et al. [34] presented the idea of detecting frame duplication by comparing graphs of frames, and similarity was checked by comparing only the spatial correlation of the original and forged clips. All these similarity detection techniques access the stored surveillance footage from the stored database; resulting in significant computation time to process each video frame. Features such as correlation [6,10,13], optical flow [17,42,43], prediction residual [22,44,45], bag-of-words (BoW) model [23], standard deviation of residual frames [40], motion vector and motion residual [36,46], and noise residue [32] have been utilized in the literature to identify inconsistencies introduced by inserted, deleted, or duplicated frames in videos. Some methods, like those of Wang et al. [11] and Huang et al. [47], use statistical features like the consistency of correlation coefficients of gray values (CCCoGV) and triangular polarity feature classification (TPFC) to detect inter-frame tampering. These algorithms, although based on statistical features, may struggle to detect tampering in the presence of different compression types. Motion residual, optical flow, and/or prediction residual based features are employed by Jia et al. [17], Kingra et al. [43], Shanableh et al. [48], Chao et al. [49], and Feng et al. [50] to detect video inter-frame tampering. When the video is tampered with, it also disturbs the texture of the video frames. This change in texture provides clues to detect forgery.

Recently, Shehnaz and Kaur detected and localized multiple inter-frame forgeries in a video by employing a histogram of oriented gradients (HoG) and local binary pattern (LBP). However, this method cannot localize frame duplication and frame shuffling attacks. Many other authors, such as Zhang et al. [13], Liao and Huang [51], Zhao et al. [52], Bakas et al. [53], Kharat et al. [15], and Shehnaz and Kaur [54], utilized texture features to detect inter-frame tampering in a video. These techniques yield good results; however, these methods are computationally extensive due to their high dimensional features. In the deep learning-based methods, Longet et al. [29] used a C3D network to detect and localize frame dropping from a single video clip, comprising 16 frames, by checking the center of the clip, i.e., between the 8th and 9th frames. They defined the confidence score with a peak detection trick and a scale term based on the output score curves to reduce false alarms. They also proposed a coarse-to-fine deep learning approach [8] for the detection and localization of frame duplication at the video and frame levels. Each video was split into 64 frames, with an overlap of 16 frames, and deep spatiotemporal features were extracted using a pretrained I3D network. Additionally, a Siamese network based on ResNet was utilized to verify frame duplication at the frame level. Location of tampering is determined with an I3D-based inconsistency detector.

Bakas et al. [27] proposed a deep learning technique based on 3D-CNN to detect inter-frame forgeries within a single video. They introduced a difference layer (pixel-wise difference layer) at the beginning of C3D, which extracts the temporal information suitable for detection of inter-frame anomalies in a video.

For detection of inter-frame tampering, Fadl et al. [21] used a pre-trained 2D-CNN model for automatic feature extraction. They computed the spatiotemporal average of every 10 non-overlapped frames of video before passing them to 2D-CNN. Then, the structural similarity index (SSIM) among features is computed, which is then fed to MSVM for classification. It detects frame insertion, deletion, and duplication, with average accuracies of 99.9, 98.7, and 98.5, respectively, but it cannot detect tampering involving fewer than 25 frames. It shows effectiveness in detecting tampering when the selected frames for insertion, deletion, or duplication are in multiples of 10. This method works with the assumption that frames should only be inserted at the static portions of the video when performing frame duplication tampering. The localization of the tampered region is not precise. Additionally, the method has shown good performance on their developed dataset, but it is not validated across different datasets, potentially limiting its generalizability.

Considering video tampering detection as an anomaly detection task, integrating deep learning techniques with prior information could potentially enhance the efficacy of video tampering detection [55]. Kumar and Gaur [56] proposed a method that extracts deep features using a CNN model. This method establishes the relationship between consecutive frames by calculating the inter-frame correlation coefficient. The inter-frame correlation distance is then computed, and a dual-threshold is applied to identify the forgery and its type. A detection accuracy of 86.5% is achieved for the VIFFD dataset. However, the VIFFD dataset used in this approach shows a lack of realistic representation, i.e., it only incorporates certain scenarios: frames are only inserted at the start or end of the video in the frame insertion forgery; if frames are removed at the beginning of the video, the deleted frames are replaced by black frames, which is not practical approach.

Deep learning techniques rely heavily on large datasets to automatically extract the high-dimensional features essential for video tampering detection. Numerous researchers have carried out experiments on their developed datasets [17,21,25,29,40,43,52,54,57], yielding commendable detection accuracies; however, these datasets are not accessible to other researchers. A thorough analysis reveals that most of the existing inter-frame tampering detection methods are based on handcrafted features, which are sensitive to post-processing operations like blurring, noise, and compression. Most of them cannot detect and localize all kinds of inter-frame tampering. The method of Long et al. [29] can detect frame deletion tampering only from the center of a 16-frame video shot. In Ref. [27], the authors construct individual trained models for each type of inter-frame tampering within a single video;

this method cannot identify frame duplication involving more than 20 frames. On the other hand, the method in Ref. [21] cannot detect tampering if the number of tampered frames are fewer than 25.

Table 1 provides comprehensive details of the relevant literature regarding inter-frame forgery detection techniques for digital video. Notably, most of the state-of-the-art deep learning-based techniques [8,29,38] can only detect a single type of temporal tampering within a video. Furthermore, they are computationally extensive due to high dimensional features. Similarly, many temporal tampering detection methods exhibit robust performance on a specific set of videos but struggle to replicate such results on other unknown video datasets.

To evaluate the effectiveness of the proposed techniques, it is necessary to test them on publicly available datasets. Unfortunately, this is a big challenge, as these datasets are not accessible to other communities or researchers. Compared to tampered image datasets tampered video datasets are significantly less mature. In this paper, we propose a novel method for detecting inter-frame tampering with high accuracy in regards to both detection and localization. Our method remains effective, even when the video exhibits variations in format, resolution, and frame rate and contains as few as 10 tampered frames. We employed 2D-CNN for feature extraction and utilized an autoencoder to reduce the computational overhead by shrinking the dimension of the feature space. LSTM/GRU, with an FC layer, is used to train all tampering simultaneously for classification. Our system overcomes the previous method's drawbacks and improves performance. The detail of the proposed method is presented in the next section.

Table 1. Analysis of inter-frame tampering detection techniques (precision: P; recall: R; detection accuracy: DA; localization accuracy: LA).

Author, Year	Method/Features	Forgeries Identified	Dataset Size	Results				Merits/Demerits
				P (%)	R (%)	DA (%)	Other	
Kingra Aggarwal et al., 2017 [43]	Prediction residual and OF	Insertion, Deletion, Duplication	Personal dataset	-	-	80/83/75	LA: 80% (for all)	Performance decreases on high illumination videos
Long et al., 2017 [29]	C3D	Deletion	Develop personal dataset	-	-	98.15	AUC: 96%	Detect single type forgery from fixed-length GOP
Zhao, Wang et al., 2018 [52]	SURF feature with FLANN	Insertion, Deletion, Duplication	10 video shots	98.07 (for all)	100 (for all)	99.01 (for all)	-	Poor localization and dataset is very small
Huang, Zhang et al., 2018 [57]	Wavelet packet decomposition, DCT	Insertion, Deletion	Personal dataset using 115 videos of SULFA, OV, 124 self-recorded	0.9876 (for all)	0.9847 (for all)	-	-	Audio data is required with videos, poor localization
Jia, Xu et al., 2018 [17]	OF and correlation between frames	Duplication	Personal dataset using videos of VTL, SULFA, DERF	0.98	0.985	-	-	Unable to detect tampering with static scenes
Bakas et al., 2018 [27]	3D-CNN	Duplication, Insertion, Deletion	9000 videos from UCF101	-	-	97%	-	Cannot detect duplication of more than 20 frames
Long, Basharat et al., 2019 [8]	I3D with Resnet152	Duplication and Localization	Media forensics challenge-18, VIRAT: 12, IPhone-4: 17	-	-	-	AUC: 84, 81.46	Detect only one type of tampering, poor localization
Fadl et al., 2021 [21]	2D-CNN with MSVM	Insertion, Deletion, Duplication	Personal dataset of 13,135 videos taken from VIRAT, SULFA IVY, and LASIESTA	-	-	99.9 98.7 98.5	-	Can detect tampering if tampered frames are in multiples of 10.
Alsakar et al., 2021 [25]	Correlation with third-order tensor tube-fiber mode	Insertion, Deletion	Personally developed forged dataset from 18 videos of TRACE library	96 92	94 90	-	F1 score: 95 91	Identification of frame duplication forgery has not been tackled
Kumar and Gaur, 2022 [56]	Inter-frame correlation distance	Insertion, Deletion	90 videos from VIFFD dataset	74	73	72	F1 score: 73	Poor DA, lack of realistic representation of dataset
Panchal et al., 2023 [58]	Video quality assessment, multiple linear regression	Single and Multiple Deletion	Personal dataset from 80 videos of VTD, SULFA, UCF-101, and TDTVD	-	-	96.25%	-	Can detect only one type of tampering
Shehnaz and Kaur, 2024 [54]	HoG with LBP	Insertion, Deletion, Duplication	Personal dataset by taking videos from SULFA, VTD	99.4	99.2	99.6	F1: 99.5	Unable to localize frame duplication tampering

3. Proposed Method

This research aims to develop an automated method that detects, localizes, and accurately and precisely determines the type of inter-frame tampering in videos. First, we formulate the problem and then present the details of the proposed method to tackle the obstacle.

3.1. Problem Formulation

We are given a surveillance video $x \in R^{r \times c \times t}$, comprising t frames (i.e., t represents the time axis), each with a resolution of $r \times c$. It is required to determine whether the video is tampered with by inserting or deleting frames. If the video is found to be tampered with, then the inserted/deleted region is located.

Let $Y = \{authentic, insetion, deletion\}$. We formulate this detection problem as a three-class classification problem and design a classifier $f : R^{r \times c \times t} \rightarrow Y$ such that $f(x;\theta) = y$, where $x \in R^{r \times c \times t}$, $y \in Y$ and θ represent the learnable parameters. This means that we need to design a mapping $f(x;\theta)$ that takes a video x as input and predicts whether it is authentic or tampered with, by insertion or deletion.

Given a video $x' \in R^{r \times c \times t}$, which is already detected as tampered with by insertion or deletion, it is required to determine the exact location of tampering. Let T represents the objective function for locating inter-frame tampering. This takes tampered video x' as the input and pinpoints the tampered region $T(x') = d$, where d precisely indicates the location of tampering within a video.

3.2. Proposed Method for Detection

We design the mapping f as a composition of four mappings, as follows:

$$f(x;\theta) = \phi_4 \circ \phi_3 \circ \phi_2 \circ \phi_1(x), \tag{1}$$

where the mapping ϕ_1 preprocesses x to yield $z_1 \in R^{r \times c \times t}$, ϕ_2 takes z_1 as input and extracts t temporal features $z_2 \in R^{m \times t}$, each of dimension m, ϕ_3 reduces the dimension m of the feature space to d and yields $z_3 \in R^{d \times t}$, and finally, ϕ_4 analyzes the temporal features z_3 to predict the label y of x.

An overview of the proposed method is presented in Figure 2; it comprises four blocks. The first block models ϕ_1 and is concerned with preprocessing; a video is split into overlapped clips, and spatial and temporal information is fused to generate a unified image for each clip. The second block specifies ϕ_2, which extracts the features. The third block defines ϕ_3, which deals with reducing the dimensionality of the feature space and generating a consolidated feature matrix for the entire video. The final block models the mapping ϕ_4, that performs the inference of whether x is authentic or tampered with via deletion or insertion. Further details are provided in the subsequent sections.

3.2.1. Preprocessing

The tampered videos exhibit inconsistencies in pixel values between two consecutive frames, as presented in Figure 1. It is challenging to detect these small variations in pixel values, especially in the case of sophisticated video tampering, where the inconsistencies are minimal. To address this, we introduced an efficient preprocessing method. Processing the input video as a whole is time-consuming and impractical for locating the tampered regions. Therefore, first, we segment the input video x into overlapping clips $\{c_1, c_2, c_3, \ldots c_n\}$, where each clip consists of L frames, i.e., $c_i = (f_{i1}, f_{i2}, \ldots, f_{iL}) \in R^{r \times c \times L}$. Subsequently, inspired by the approach provided in Ref. [59], we pool the spatial and temporal information corresponding to each clip, yielding spatiotemporal pooling (STP). For pooling, first, each frame of a clip c_i is filtered using the 3×3 averaging filter, i.e.,

$$\widetilde{c_i} = \psi_1(c_i), \tag{2}$$

where ψ_1 filters each frame f_{ij} of c_i to generate $\widetilde{c}_i = \left(\left(\widetilde{f}_{i1}, \widetilde{f}_{i2}, \ldots, \widetilde{f}_{iL}\right)\right)$. Then, $STP_i \in R^{r \times c}$ of c_i is computed by averaging the frames of $\widetilde{c}_i$, as follows:

$$STP_i = \psi_2(\widetilde{c}_i) = \frac{1}{L}\sum_{j=1}^{L} \widetilde{f}_{ij}\,. \tag{3}$$

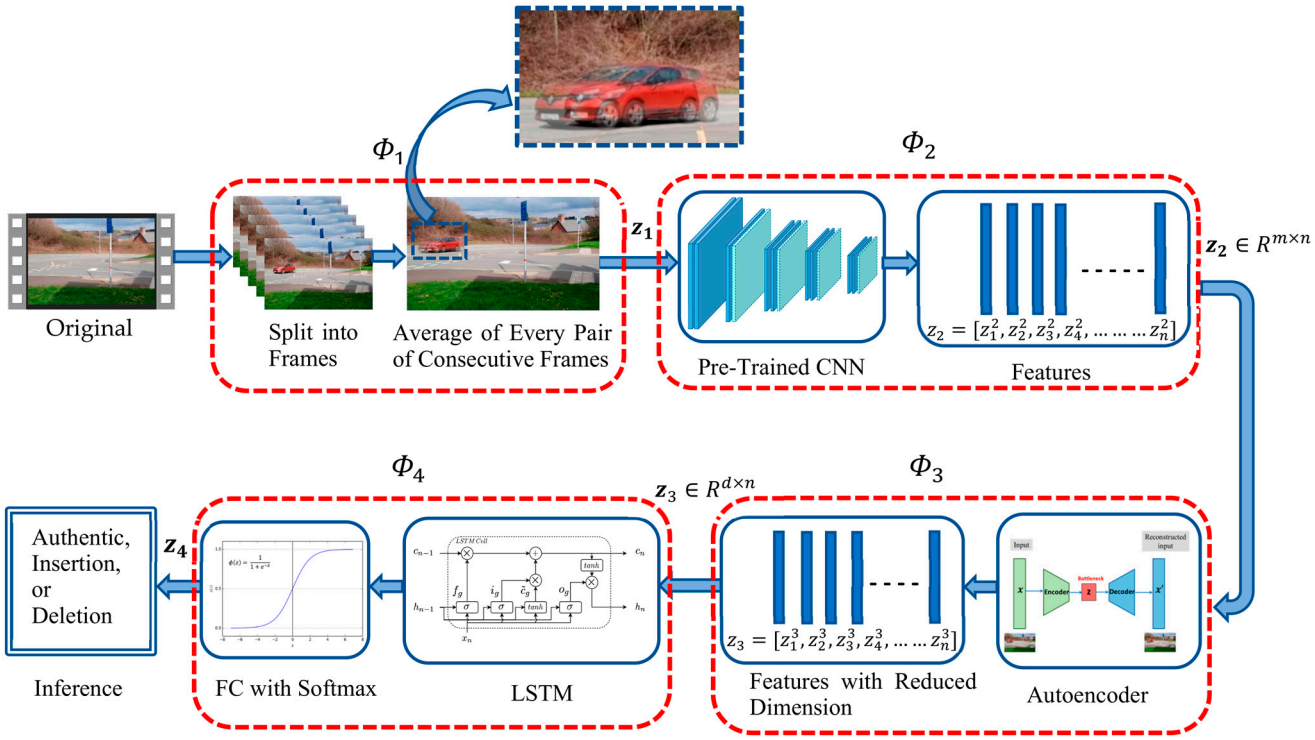


Figure 2. Workflow of the proposed method for inter-frame tampering detection in a video. First, a video is split into frames, and the average of every consecutive pair is computed. The average frames are passed to a CNN for the extraction of features, which are fed to an autoencoder to reduce dimension, and then to an LSTM. The output of the LSTM is finally passed to an FC layer, with softmax activation, that yields the posterior probabilities of the classes.

An STP (zoomed-in form) of a video clip is presented in Figure 2; it shows the movement of a red car between two consecutive frames. As we use a pre-trained CNN model for feature extraction from each STP, each STP image is resampled to the dimensions of $a \times b \times 3$ using the bicubic interpolation algorithm. Images resampled with the bicubic interpolation exhibit high picture quality, smoother textures, and fewer interpolation artefacts, resulting in significantly superior outcomes [21,60,61]. In our case $a = 224$ and $b = 224$ because we employ VGGNet for feature extraction.

Finally, the concatenation of STP_i, $i = 1, 2, \ldots, n$ results in z_1, where

$$z_1 = [STP_1, STP_2, \ldots, STP_n]. \tag{4}$$

The above operations define the mapping ϕ_1, which takes the video x and generates $z_1 \in R^{r \times c \times n}$, i.e.,

$$z_1 = \phi_1(x) = \psi_3 \circ \psi_2 \circ \psi_1 \circ \psi_0(x) \tag{5}$$

where ψ_0 is the mapping that splits the input video into n overlapping video clips, and ψ_3 represents the concatenation operation.

The number of frames L in each clip plays a key role in encoding tampering sensitivity. There are two possibilities: L is fixed for all clips, and L varies for each clip, i.e., adaptive L. For adaptive L, the video comprising t frames is segmented into clips of variable lengths, based on the adaptive selection of the number of frames until the mean difference between

the ith frame and each subsequent $i + 1$ to $i + k$ frames exceeds the specified threshold, here, $i = 1, 2, 3, \ldots, t$ and $k = 1, 2, 3, \ldots, t - 1$. Each clip starts from the frame where the mean difference exceeds the threshold, and this process persists until the video's conclusion. In this way, we obtain segmented clips $C_i = \{c_1, c_2, c_3, \ldots, c_n\}$, with variable lengths. After conducting numerous experiments, a threshold value of 100 is chosen. We discuss the effect of L in Section 5.1.

3.2.2. Feature Extraction

Videos can vary significantly in regards to content, quality, resolution, lighting conditions, camera motion, and other factors. Extracting robust features that are unaffected by such variations is challenging and requires careful consideration of feature selection and preprocessing techniques. To address this, we have introduced an efficient method that extracts discriminative features by employing advanced CNN models initially trained on ImageNet. CNNs have demonstrated effectiveness in tasks like image classification [62] and object localization [63] within images. We employed a state-of-the-art pre-trained CNN model, VGG-16, which consists of 13 convolutional layers, 5 max-pooling layers, and 3 dense layers, totaling 21 layers. However, it contains only 16 weight layers, which are learnable parameters layers. Following a stack of convolutional layers, three fully connected (FC) layers are utilized: the first two, with 4096 channels each, and the third, performing 1000-way ILSVRC classification, containing 1000 channels, one for each class. The final layer is the softmax layer. Since it is explicitly trained on millions of images, e.g., "ImageNet" [64], it is thus effective for modeling vision-related problems [65,66]. The convolutional base of this pre-trained model is used to extract rich and deep features, while discarding the remaining network layers. Additionally, we introduce a global average pooling layer after the convolutional base of VGG-16. Since the global average pooling layer has no parameter, using it instead of fully connected layers significantly reduces the model's complexity. Each STP image, $224 \times 224 \times 3$ in size, is fed to VGG-16 to get feature vector u_i of size m, i.e.,

$$U = \mathfrak{G}_0(z_1) = [u_1, u_2, u_3, \ldots u_n], \tag{6}$$

where $\mathfrak{G}_0$ extracts feature matrix U of size $m \times$ n from the resampled STPi images z_1, i.e., $U \in R^{m \times n}$. This mapping is represented as:

$$z_2 = \phi_2(z_1) = \mathfrak{G}_0(z_1), \tag{7}$$

where the mapping ϕ_2 takes z_1 as input and generates a feature matrix $z_2 \in R^{m \times n}$. Since the global average pooling layer generates a feature vector of size 512, corresponding to each STP image, the feature matrix $512 \times$ n is obtained for each video.

3.2.3. Dimensionality Reduction

High-dimensional features are prone to overfitting, require more memory, and can lead to computational difficulties. Many approaches have high computational complexity due to high dimensional features [35–37], which is a significant task. Similarly, choosing an appropriate dimensionality reduction method is also challenging. Principal component analysis (PCA) simplifies data using linear combinations, while the use of autoencoders, a more flexible approach, captures non-linear patterns, making them powerful for complex data reduction, with enhanced representation capabilities. The nonlinear dimensionality reduction technique using autoencoders can effectively learn the nonlinear correlation among numerous variables and succeed in detecting anomalies. In contrast, linear PCA, which employs linear dimensionality reduction, overlooks anomalies [67]. To address this issue, we proposed the use of an autoencoder to reduce the dimensionality of feature space in an unsupervised way [68,69]. This autoencoder was developed with a single hidden layer, using ReLU as the activation function and Adam as an optimizer. The encoder takes deep features and maps them into a latent encoding space, generating a latent code; the decoder then takes the encoder's output and attempts to reconstruct the original input,

as shown in Figure 3b. We adopted this simple architecture because networks with more hidden layers are difficult to train [70]. We employed an autoencoder $\pounds_0 : R^{m \times n} \rightarrow z_3$ such that $\pounds_0(z_2, \alpha) = V$, where $z_2 \in R^{m \times n}$ and $V \in R^{d \times n}$, and α represents the learnable parameters, i.e.,

$$\pounds_0(z_2, \alpha) = V = [v_1, v_2, v_3, \ldots v_n], \tag{8}$$

where an autoencoder takes each feature vector u_i of z_2 with dimension m and mapped to a vector v_i of dimension d, where $d << m$. This mapping is represented as

$$z_3 = \phi_3(z_2) = \pounds_0(z_2, \alpha), \tag{9}$$

where ϕ_3 takes feature matrix $z_2 \in R^{m \times n}$ as input and generates a feature matrix $z_3 \in R^{d \times n}$ of reduced dimensionality, corresponding to a video.

The reduction in dimensionality of feature space not only leads to notable reductions in training time, memory usage, and computational overhead, but also enhances the visibility of tampering traces. Figure 3 represents an example of frame deletion tampering when video clip frames from 107 to 121 have been removed. The effect of CNN features, without using an autoencoder, is presented in Figure 3c, and after using autoencoder in Figure 3d, where the feature dimension is reduced from 512 to 128 per STP image.

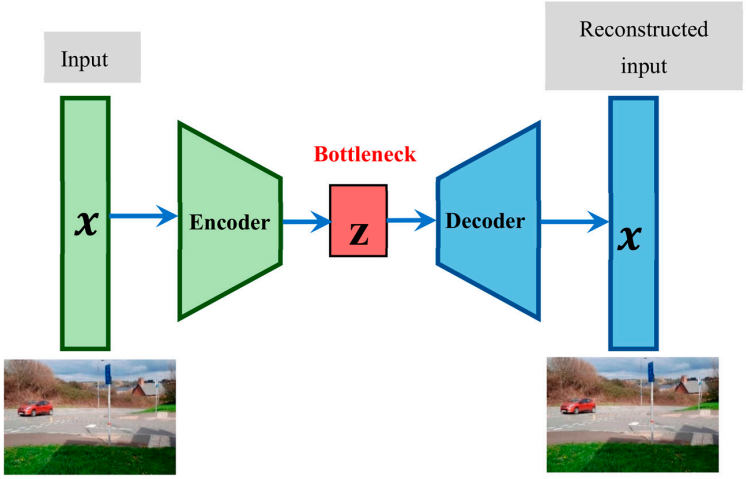
3.2.4. Classification

Video frames often exhibit temporal dependencies, where information in one frame may be related to information in temporally distant frames. Traditional neural networks may struggle to effectively capture such long-range dependencies. Hochreiter and Schmidhuber introduced LSTM networks [71] and GRUs [72] that can analyze long-range dependencies among video frames and detect tampering traces. For classification, we employed LSTM/GRU with an FC layer, activated through the softmax function, to compute the probability of the video sequence being categorized as $y \in \{authentic, insetion, deletion\} = Y$. This classifier is represented as $E : R^{d \times n} \rightarrow Y$ such that $E(z_3; \beta) = y$, where $z_3 \in R^{d \times n}$, $y \in Y$, and β represent the learnable parameters. This classifier operates on a reduced feature matrix z_3 and assigns it to one of three categories: authentic, insertion, or deletion.

$$z_4 = E(z_3; \beta) = y. \tag{10}$$

This mapping is represented as

$$z_4 = \phi_4(z_3). \tag{11}$$



(**a**)

Figure 3. *Cont.*

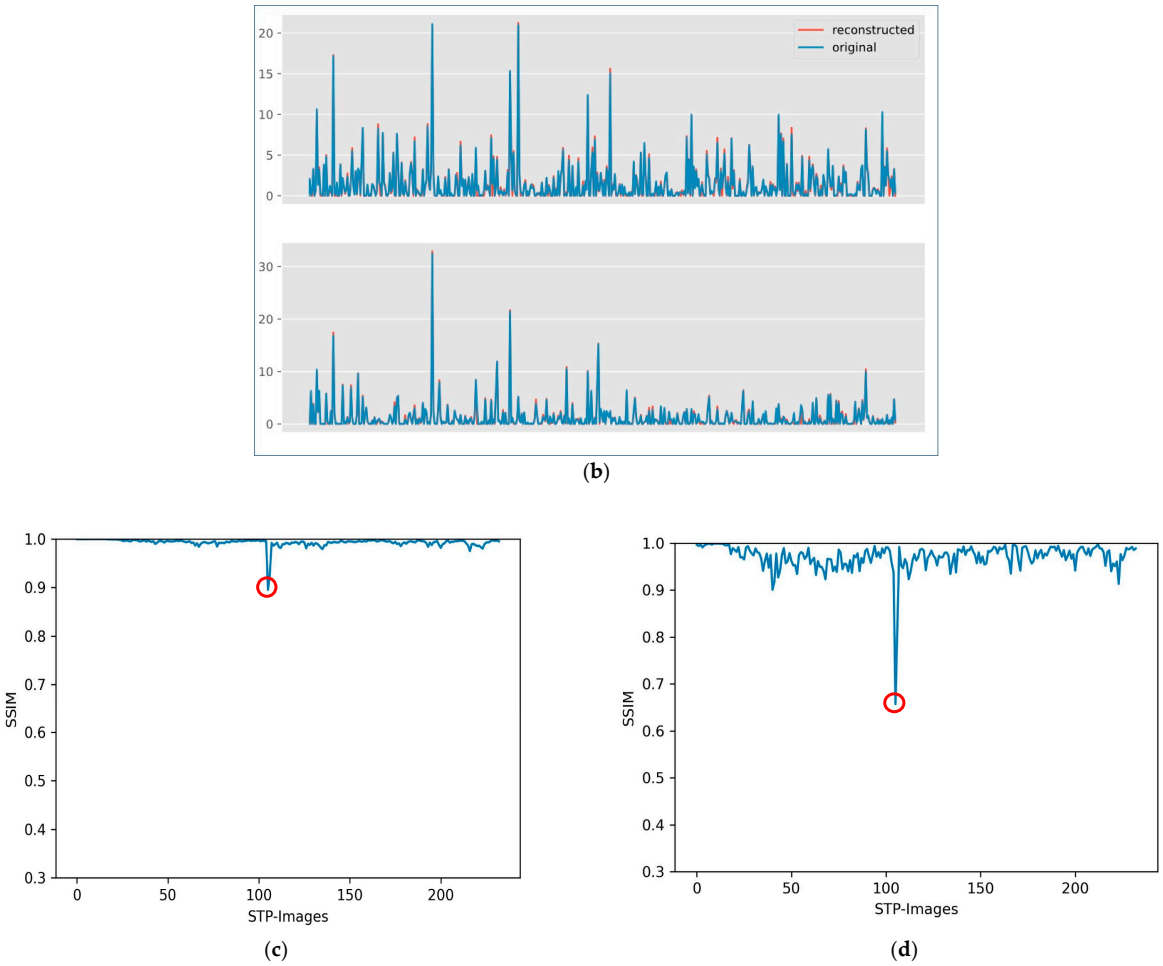


Figure 3. Effect of using an autoencoder for reducing the dimensionality of feature space, the tampering point is indicated by falling peak marked with a red circle (Door Opening(N).mp4). (**a**) Autoencoder. (**b**) Reconstructed features. (**c**) Before dimensionality reduction. (**d**) After dimensionality reduction.

It demonstrates that reduced feature z_3 obtained from the previous block ϕ_3, as shown in Figure 2, is fed to LSTM/GRU, which further analyses the tampering traces, and a final inference z_4 corresponding to the entire video is generated. This mapping is represented by Equation (11). In this way, classification is conducted at the video level. The functions of LSTM and GRU are described below.

LSTMs efficiently improve performance by memorizing the relevant information for a long period of time and identifying the pattern through its gates. The LSTM layers also examine the inter-frame inconsistencies in the extracted features, leading to high accuracy in detecting tampering in videos, irrespective of their frame rates, video formats, number of tampering frames, and compression quality. The standard formulation of a single LSTM cell is described by the following equations:

$$f_n = \sigma\Big(W_f.[h_{n-1}, z_{3n}] + b_f \Big), \tag{12}$$

$$i_n = \sigma(W_i.[h_{n-1}, z_{3n}] + b_i), \tag{13}$$

$$\widetilde{C}_n = tanh(W_c.[h_{n-1}, z_{3n}] + b_c), \tag{14}$$

$$C_n = f_n * C_{n-1} + i_n * \widetilde{C}_n, \tag{15}$$

$$o_n = \sigma(W_o.[h_{n-1}, z_{3n}] + b_o), \tag{16}$$

$$h_n = o_n * \tanh(C_n), \tag{17}$$

where σ is the sigmoid function, tanh is the hyperbolic tangent function, and i, o, f, C, and $\widetilde{C}$, are the input gate, output gate, forget gate, memory cell content, and new memory cell content, respectively.

GRUs were introduced in 2014 [72], and are similar to LSTM but they are fast, compact, more efficient in terms of simpler structure, and have fewer parameters [73]. GRU formulation is given by the following equations

$$r_n = \sigma(\ W_{zr} z_{3n} + W_{hr} h_{n-1} + b_r), \tag{18}$$

$$e_n = \sigma(\ W_{xe} z_{3n} + W_{he} h_{n-1} + b_e), \tag{19}$$

$$\widetilde{h}_n = tanh\Big(\ W_{zh} z_{3n} + W_{hh}\Big(r_n \bigodot h_{n-1}\Big) + b_h\Big), \tag{20}$$

$$h_n = e_n \bigodot h_{n-1} + (1 - e_n) \bigodot \widetilde{h}_n, \tag{21}$$

where σ represents sigmoid activation, W represents weight matrices, b represents biases, tanh is the hyperbolic tangent, and z_{3n}, r_n, e_n, and h_n are the input vector, reset gate, update gate, and output vector, respectively.

These recurrent neural networks excel at capturing temporal dependencies in sequential data, making them applicable to tasks involving video analysis and tampering detection. Both approaches utilize a different method of fusing previous time step information with gates to prevent vanishing gradients.

To calculate the loss of the video tampering classification model, the cross-entropy loss function is applied, as provided below:

$$Loss = -\sum_{i=1}^{output\ size} y_i.log\ \hat{y}_i, \tag{22}$$

where $\hat{y}_i$ is the predicted score of class i at the softmax layer.

3.3. Proposed Method for Localization

Once the video is detected as tampered with, along with its type, the subsequent task involves pinpointing the tampered region to gain more trust of the end user. For localization, first, the SSIM among the features of reduced dimension is computed, i.e.,

$$S = \beta(z_3) = [s_1, s_2, s_3, \ \ldots s_{n-1}], \tag{23}$$

where $z_3 \in R^{d \times n}$ contains a feature matrix of reduced dimensionality, as presented in Section 3.2.3, β computes the SSIM among consecutive features, and S contains a sequence of all the SSIM values of the tampered video.

SSIM utilizes the luminance, contrast, and structural element information for the compared images. Tampering may result in abrupt changes in SSIM values or inconsistencies, where three scenarios are perceived. In the first scenario, no dominating falling points are detected in the SSIM curve, indicating that there are no inconsistencies. Such videos are classified as original; examples of this scenario are illustrated in Figure 4a–c. In the second scenario, a single obvious falling point is present, leading to the classification as tampered, by frame deletion attack. This falling point denotes the precise location of frame deletion tampering, as depicted in Figures 4d–f and 4g–i, for feature dimensions of 512 and 128, respectively. In the third scenario, two distinct falling points are observed, resulting in the classification as tampered, by frame insertion attack. These falling points represent the start and end points of frame insertion tampering, as demonstrated in Figures 4j–l and 4m–o, for feature dimensions of 512 and 128, respectively. In this way, SSIM effectively localizes

video tampering by identifying regions where the spatiotemporal characteristics of the video have been altered.

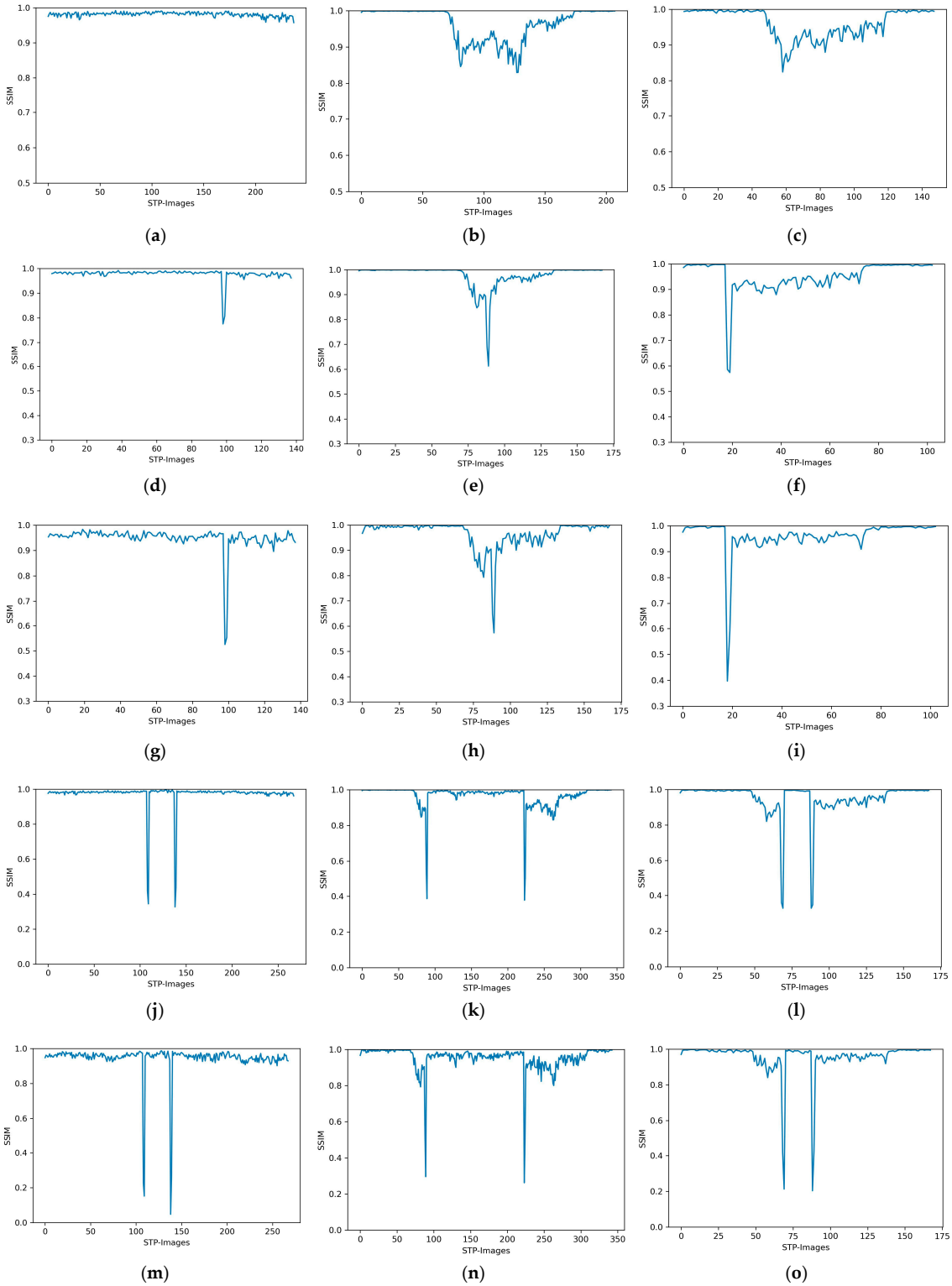


Figure 4. Examples of tampering localization of different cases; the first row (**a**–**c**) shows SSIM of features of original videos; the second row (**d**–**f**) shows SSIM of features of frame deletion tampering, without dimensionality reduction; the third row (**g**–**i**) shows SSIM of features of frame deletion tampering with dimensionality reduction to 128; the fourth row (**j**–**l**) shows features of frame insertion tampering, without dimensionality reduction; the fifth row (**m**–**o**) shows features of frame insertion tampering with dimensionality reduction to 128.

In the case of video with deletion tampering, the lowest value of S represents the point of deletion and is represented as:

$$M(s_k) = p, \text{ where} k = 1, 2, 3, \ldots, n-1, \tag{24}$$

where M takes the sequence S as input and determines the lowest SSIM value p, representing a falling peak, and the value of k at p represents the exact position of frame deletion. In the case of video with insertion tampering, the two lowest values of sequence S, i.e., p and q, represent the two falling peaks and are represented as:

$$M(s_k) = p \text{ and } q. \tag{25}$$

The values of k at p and q represent the tampered locations, i.e., the smaller value of k denotes the start, and the larger value of k indicates the end point of insertion within a video.

4. Dataset Description and Evaluation Protocols

Dataset and Evaluation Protocols

Due to the absence of extensive video datasets for detecting inter-frame tampering in surveillance videos [74], we developed the COMSATS Structured Video Tampering Evaluation Dataset (CSVTED) that contains challenging videos with different complexity levels, i.e., from simple background to complex background, single object to random movement of multiple objects, and different lighting conditions such as morning, noon, evening, night, and fog. The videos are recorded by multiple cameras of different models, capturing both moving and static views; CSVTED covers all forms of tampering, including frame deletion, insertion, duplication, copy-move, and splicing. The videos in CSVTED have frame rates ranging from 12 to 30 fps and duration spanning from 5.648 to 75 s. Moreover, the videos in CSVTED portray natural scenes post-tampering and are available in popular formats such as avi, mp4, or mov. Illustrations of some frames from CSVTED are provided in Figure 5. Extensive experiments were carried out on numerous videos captured by the static cameras, varying the number of inserted/deleted frames per test video from 10 to 545. To evaluate the performance of our system, we used 10-fold cross-validation. Further details of the developed dataset are presented in Table 2.

For evaluation, we used standard metrics, including precision rate (PR), recall rate (RR), F1 measure, sensitivity (TPR), specificity (TNR), and detection accuracy (DA), which are given by

$$\text{PR} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \text{ RR} = \frac{\text{TP}}{\text{TP} + \text{FN}}, \text{ F1} = \frac{2 \times \mathit{Precision} \times \mathit{Recall}}{\mathit{Precision} + \mathit{Recall}}, \tag{26}$$

$$\text{TPR} = \frac{\text{TP}}{\text{TP} + \text{FN}}, \text{ TNR} = \frac{\text{TN}}{\text{TN} + \text{FP}}, \text{ DA} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}, \tag{27}$$

where TP represents the count of true positives (i.e., accurately identified instances of video forgeries); FP represents the count of false positives (i.e., incorrectly identified instances of video forgeries or falsely identified forgery type); TN represents the count of true negatives (i.e., accurately identified instances of video authentication); and FN represents the count of false negatives (i.e., incorrectly identified instances of video authentication). The method is deemed satisfactory when achieving high detection rates for TPR, TNR, and DA.

Table 2. Details of datasets used in the literature for inter-frame tampering detection.

Reference	Total Videos Used in the Experiment	Number of Frames Duplicated	Resolution	Frame Rate	Scenario
Ulutas et al., 2018 [23]	31	20, 30, 40, 50, 55, 60, 70, 80	320 × 240	29.97, 30	-
Jia et al., 2018 [17]	115	10, 20, 40	320 × 240	29.97, 30	-
Ulutas et al., 2017 [75]	10	20, 30, 40, 50, 55, 60, 70, 80	320 × 240	29.97, 30	-
Fadl et al., 2021 [21]	869 (FI: 287 + FD: 420 + Dup: 62 + Ori: 100)	10 to 600	1280 × 720, 320 × 240, 352 × 288, 704 × 576	23.98 to 30	-
CSVTED	2555 (FI:1326 + FD: 981 + Ori: 248)	10 to 545	640 × 360, 640 × 480, 1920 × 1080, 1280 × 720,	12.50, 15, 25, 29.97, 30	Morning, Noon, Evening, Night, and Fog

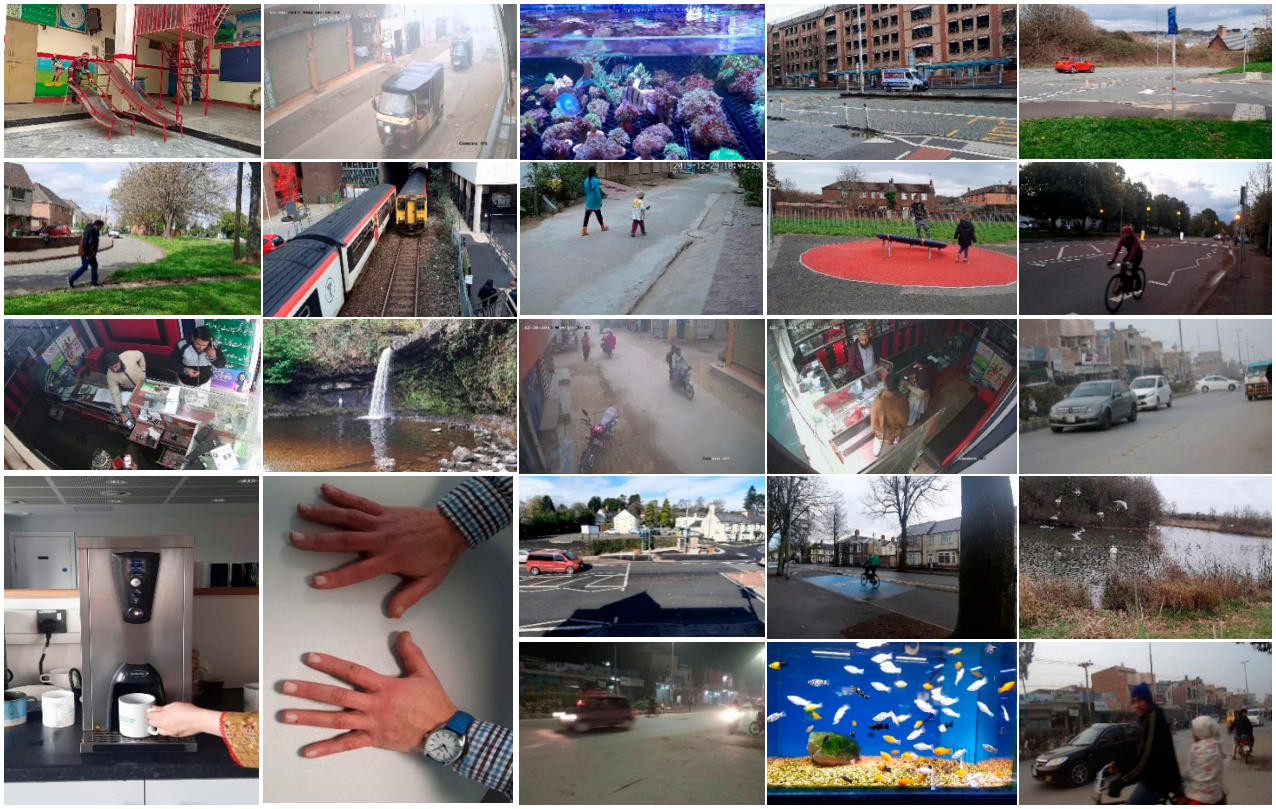


Figure 5. Sample frames from CSVTED.

5. Experimental Results and Discussion

Extensive experiments are carried out on a notebook computer equipped with an RTX2070 graphics card, a core i7 processor, and 32 GB RAM, running Python 3.8.12. Different libraries used for the experimentation are OpenCV, Keras, sklearn, Tensorflow, Pandas, and matplotlib. VGG16 with LSTM and GRU are used to perform experiments. Cross entropy loss, batch size of 32, learning rate of 0.001, Adam optimizer, and early stopping with patience 50 are employed to train the neural network models. Precision, recall, F1-score, TPR, and detection accuracy metrics are used for model evaluation. A statistical paired *t*-test is applied to compare the averages/means and standard deviations of the LSTM/GRU models with different configurations to determine if there is a significant difference between these models. In this test, we conducted two replications of 10-fold

cross-validation. For each replication, the available data are randomly divided into ten equal-sized sets. Each learning algorithm is trained on nine sets and tested on the remaining set. The experiments were run using two models, as previously described: standard LSTM and GRU, each with four different configurations. The details of the layers, along with the number of neurons in each layer, are provided in the subsequent sections.

5.1. The Effect of L

The choice of the number of frames L in a clip has a significant impact on the performance of the proposed method. To show its effect and choose the best L, we selected a video x (Moving Fish 01.mp4), which is tampered with by deletion, and computed the corresponding $z_1 = \phi_1(x)$ and extracted features matrix $z_2 = \phi_2(x)$. Using z_2, we computed the SSIM [76] among features. Figure 6 shows the SSIM curves for different choices of L.

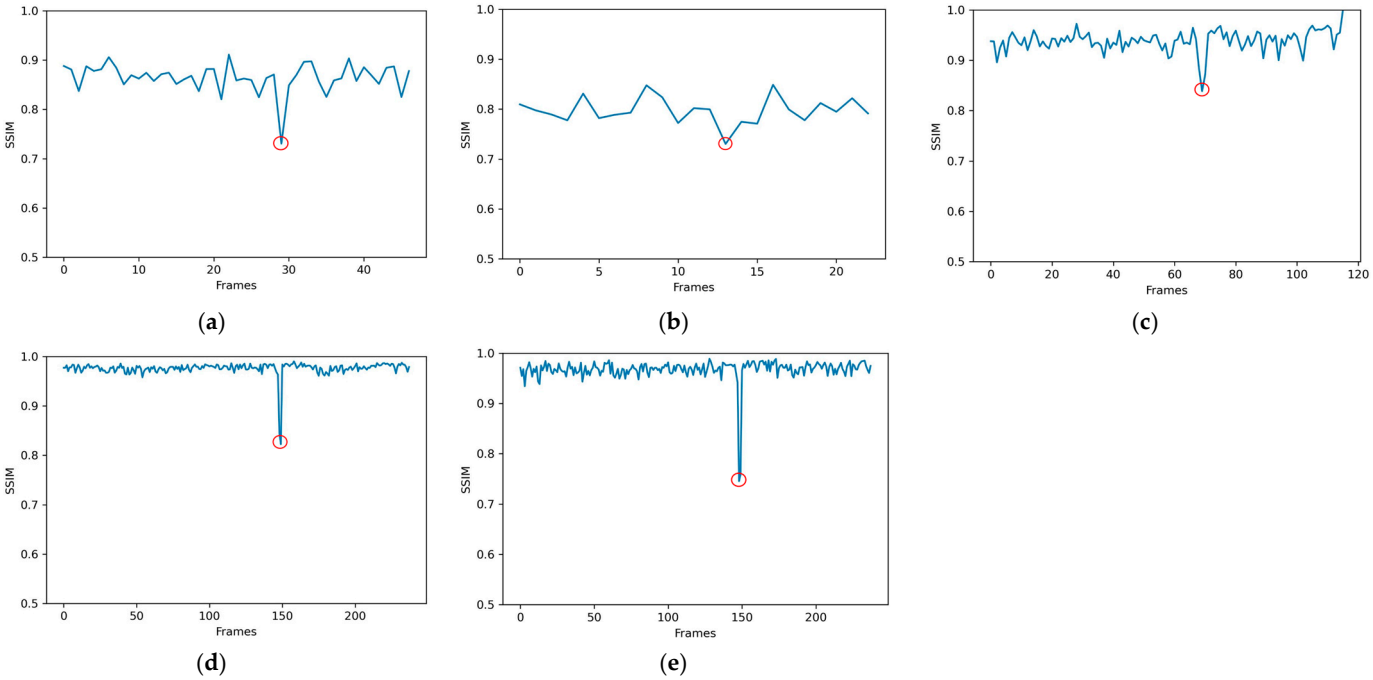


Figure 6. The SSIM curves showing the effect of various choices of L for deletion, falling peaks marked with a red circle represent tampering points (Moving Fish 01.mp4). (**a**) L = 05. (**b**) L = 10 [21]. (**c**) Adaptive L. (**d**) L = 02. (**e**) L = 02 with reduced feature dimension.

While the large value of L, i.e., 10, as used by Sondos Fadl and Qi Han et al. [21], reduces the temporal redundancy due to the slight differences among consecutive frames, it results in the loss of tampering traces, as shown in Figure 6b; L = 5 is relatively better than L = 10, but it also does not result in as sharp peak, as in the case of L = 2. Further, adaptive L also does not show better performance than L = 2 in pinpointing the deletion spot in the video. L = 2 preserves the temporal coherence in a better manner and detects the deletion region in the video, consistent with the findings by Yoo and Chang et al. [77].

Similarly, we selected another video x (Customer Dealing.mp4), which is tampered with via insertion. After computing z_1 and z_2, the SSIM among features is also computed. Figure 7 shows the SSIM curves for different choices of L. L = 10 and L = 5 do not represent two separate peaks (one peak for the start point and the second for the end of insertion). Further, adaptive L does not show better performance than L = 2 in pinpointing the start and end point of frame insertion according to its two sharp peaks, as shown in Figure 7d. L = 2 identifies insertion regions in a better manner.

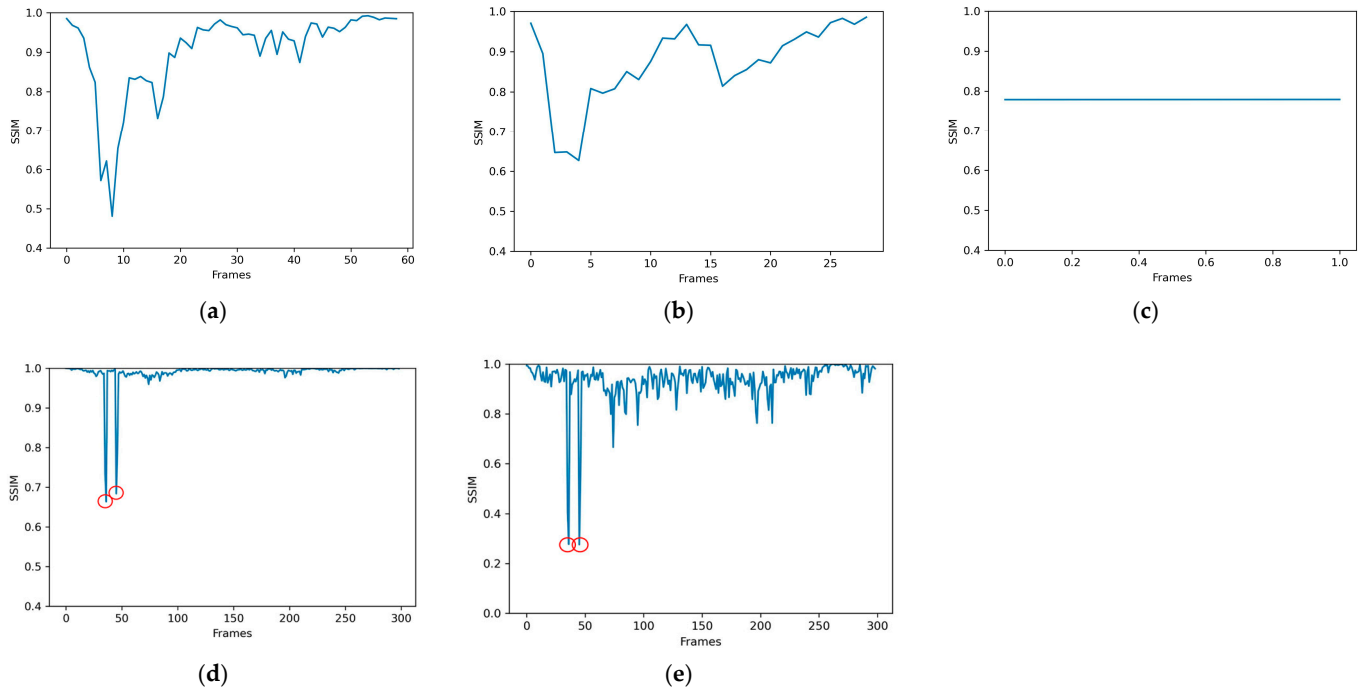


Figure 7. The SSIM curves showing the effect of various choices of L for insertion, falling peaks marked with a red circle represent tampering points (Customer Dealing.mp4). (**a**) L = 05. (**b**) L = 10 [21]. (**c**) Adaptive L. (**d**) L = 02. (**e**) L = 02 with reduced feature dimension.

In view of these observations, we set $L = 2$ in our experiments.

5.2. The Effect of Dimensionality Reduction

Dimensionality reduction reduces the redundancy among features and focuses on the most relevant and discriminative features for tampering detection, thereby enhancing detection performance. Figures 6 and 7 show the SSIM curves for frame deletion and insertion across various choices of L. While $L = 02$ yields sharp peaks, employing $L = 02$ with reduced feature dimensions results in even stronger peaks, as shown in Figures 6e and 7e. $L = 02$, with reduced feature dimensions, significantly improves the tampering traces, which yields better localization and results in less computational cost. We performed extensive experiments using features with different dimensions, and a comparison is given in Section 5.8. Utilizing dimensionality reduction proves advantageous for detecting inter-frame tampering, especially in the case of large-scale video datasets.

5.3. LSTM with Varying Depths

To assess the effectiveness of the LSTM network, we investigate the impact of various LSTM depths on classification performance. After the LSTM, an FC layer is incorporated, with softmax activation. The number of LSTM layers is incremented from 1 to 4. The number of neurons is kept to 15 in all LSTM layers. In order to avoid overfitting, early stopping is employed with patience 50, where patience is a hyper parameter. The number of iterations is set to 300. The findings indicate that a single-layer LSTM performs well when compared to multiple-layer LSTM configurations across all dimensions. Figure 8 illustrates that when the number of LSTM layers continues to increase, the network performance decreases, likely due to model overfitting. Table 3 presents the results of the paired t-test, with p values calculated at a significance level of 5%, comparing the performance of the single-layer LSTM model with that of multi-layer LSTM. Experimental results show that single-layer LSTM outperforms multi-layer LSTM.

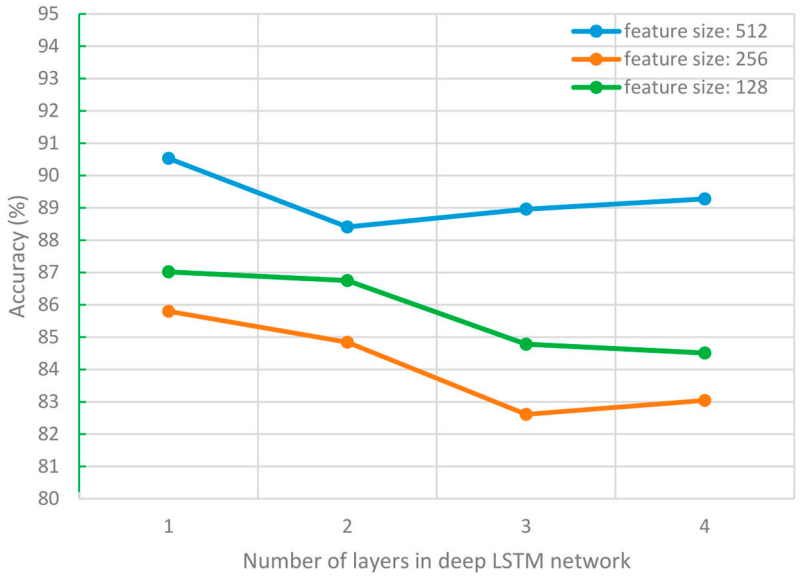


Figure 8. Deep LSTM network performance with different numbers of layers on different feature dimensions.

Table 3. *p*-Value using paired *t*-test for different deep LSTM networks.

Models	*p*-Value (At 5% Significance Level)	The Model with Significant Performance
1-layer LSTM (LSTM-L1) vs. 2-layer LSTM (LSTM-L2)	0.0003399	LSTM-L1
1-layer LSTM vs. 3-layer LSTM (LSTM-L3)	0.0038	LSTM-L1
1-layer LSTM vs. 4-layer LSTM (LSTM-L4)	0.0050	LSTM-L1

5.4. LSTM with Varying Numbers of Neurons

In this segment, the primary objective is to optimize the number of LSTM neurons while keeping the depth fixed, since single-layer LSTM shows better performance than multi-layer LSTMs. We further assess the performance of single-layer LSTM by varying the number of neurons, ranging from 15 to 150. Figure 9 depicts the classification performance of the single-layer LSTM model; the performance decreases when the number of neurons is increased above 15, in the case of 512 feature dimensions, while the performance increases in the case of 256 and 128 feature dimensions; however, it remains consistent when the number of neurons exceeds 100. Although a peak accuracy of 90.53% is achieved with 15 neurons and 512 feature dimensions, the significance of achieving an accuracy of 90.12% with the same model by utilizing 100 neurons, even with a much smaller feature size of 128, cannot be overlooked. Thus, the structure based on a single-layer LSTM (LSTM-L1) network, with 15 neurons, is suitable and recommended for inter-frame tampering detection.

5.5. GRU with Varying Depths

In order to test and verify the effectiveness of the GRU network, we explore the effects of different GRU depths on classification performance. After GRU, an FC layer, with softmax activation, is added, yielding posterior probabilities of the classes. The number of GRU layers is increased from one to four. Specifically, a one-layer GRU comprises one GRU layer with 64 neurons; a two-layer GRU comprises 8 and 64 neurons; a three-layer GRU has 8, 32, and 64 neurons; and the number of neurons used in a four-layer GRU is 8, 16, 32, and 64. Figure 10 illustrates how varying the number of GRU layers affects the detection accuracy of the network. It can be distinctly seen that there is a minor decline in

detection accuracy due to the addition of GRU layers when features with 512 dimensions are used, likely due to model overfitting, and then it remains consistent until reaching four GRU layers.

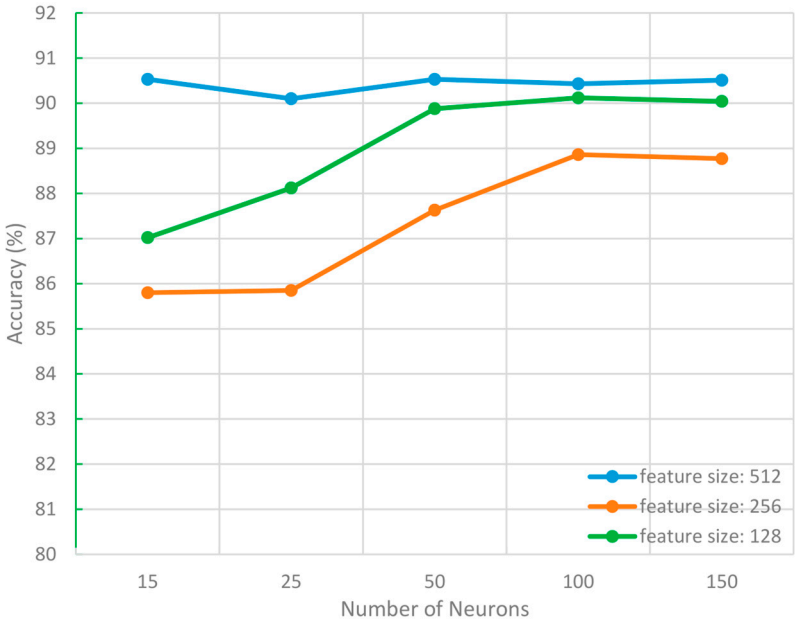


Figure 9. Classification performance of LSTM for different numbers of neurons.

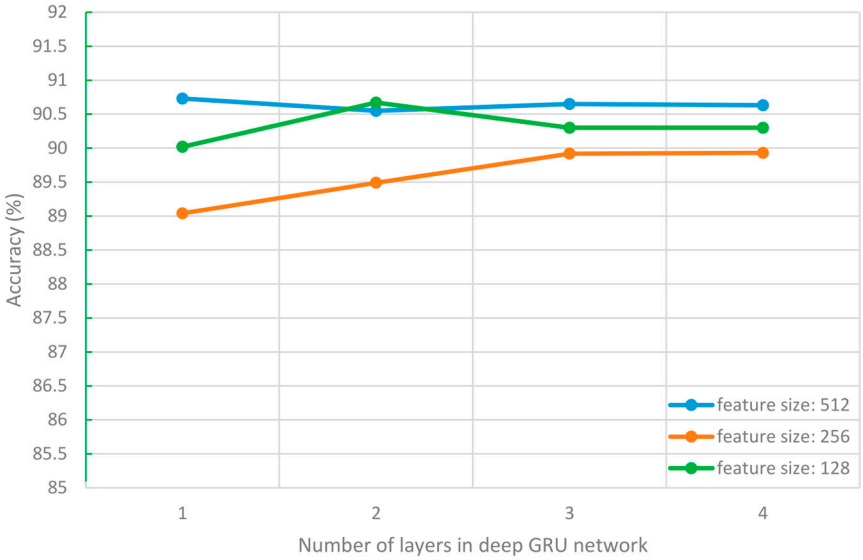


Figure 10. Deep GRU network performance with different numbers of layers on different feature dimensions.

Figure 10 reveals a clear trend in the case of features with dimensions of 256 and 128: the accuracy improves with additional GRU layers, up to the addition of three layers. This observation suggests that higher numbers of GRU layers enable the model to capture more effective features when inputs of 256 and 128 dimensions are used. However, increasing the depth beyond a certain point may lead to overfitting due to increased model complexity. The outcome indicates that the performance of GRU-L2 with a feature dimension of 128 is very close to that of GRU-L1 with dimensions of 512, thereby reducing the computation complexity and showing the effectiveness of dimensionality reduction.

We utilize a statistical paired *t*-test to compare GRU models with various configurations, aiming to ascertain any significant differences between these models. Table 4

presents the *p*-values at a 5% significance level for the comparison between single-layer and multi-layer GRUs, indicating no significant difference when GRUs with different layers are employed. However, the highest accuracy of 90.73% is achieved by a single-layer GRU with 64 neurons.

Table 4. *p*-value using paired *t*-test for different deep GRU networks.

Models	*p*-Value (At 5% Significance Level)	Model with Significant Performance
1-layer GRU (GRU-L1) vs. 2-layer GRU (GRU-L2)	0.2845	No significant difference
1-layer GRU vs. 3-layer GRU (GRU-L3)	0.4134	No significant difference
1-layer GRU vs. 4-layer GRU (GRU-L4)	0.1767	No significant difference

5.6. GRU with Varying Number of Neurons

In this segment, the primary aim is to optimize the number of GRU neurons while maintaining the fixed depth. Since the highest detection accuracy is achieved by single-layer GRU, we further analyze the performance of single-layer GRU by varying the number of neurons, i.e., spanning from 8 to 128. The experimental results revealed that performance increases when the number of neurons is increased and remains consistent when the number of neurons exceeds 64 for 512 and 128 feature dimensions. Figure 11 illustrates the detection accuracy of single-layer GRU models, highlighting the optimal performance with 64 neurons when employing features with 512 dimensions, achieving a peak accuracy of 90.73%.

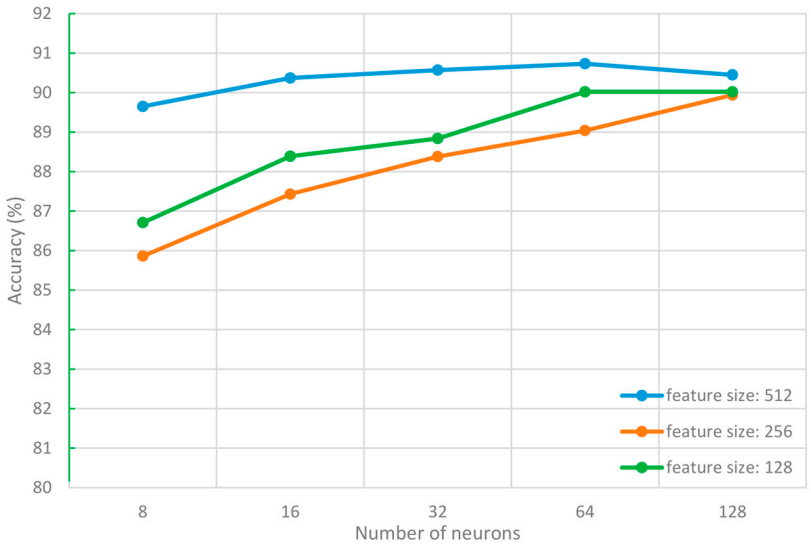


Figure 11. Classification performance of GRU for different numbers of neurons.

When comparing LSTMs and GRUs with various configurations, it becomes evident that LSTM-L1, along with GRUs ranging from L1 to L4 with a feature dimension of 512 and a GRU-L2 with a feature dimension of 128, exhibit superior detection accuracy. LSTMs are characterized by more gates and parameters, rendering them more flexible but also entailing higher computational costs and a greater risk of overfitting. In contrast, GRUs have fewer gates, fewer parameters, and require less computational power than LSTMs, making them simpler and faster.

5.7. Localization

Table 5 shows a comparison of the detection and localization accuracies of the proposed models with the state-of-the-art method for both types of tampering. The superior results of our method show its effectiveness for the detection and precise localization of frame insertion and deletion tampering.

Table 5. Detection and localization of forged videos (in %) (DA: detection accuracy; LA: localization accuracy.

Method	Tampering Type	DA	LA
Kingra et al., 2017 [43]	Insertion	80	80
	Deletion	83	73
Fadl et al., 2021 [21]	Insertion	91	91
	Deletion	60	60
Proposed Model-01 (2D-CNN with LSTM-L1)	Insertion	98.50	98.50
	Deletion	93.30	93.30
Proposed Model-02 (2D-CNN with GRU-L1)	Insertion	**98.98**	**98.98**
	Deletion	**94.18**	**94.18**
Proposed Model-06 (2D-CNN with GRU-L2)	Insertion	98.34	98.34
	Deletion	92.14	92.14

Highest values are represented in bold corresponding to each tampering.

5.8. Discussion and Comparison with State-of-the-Art

Based on the experimental results carried out, the proposed system has been compared with similar state-of-the-art methods, based on deep learning features [8,21,29,38] and handcrafted features [43,56]. A C3D-based method in Ref. [29] only detects the frame-dropping forgery at the center of a single video clip of 16 frames, while the deletion may not necessarily be in the center; moreover, it is unable to identify other forms of inter-frame tampering. Unlike existing methods [8,29,38] that focus on detecting only one type of video tampering, the proposed approach can detect both types (i.e., insertion and deletion) simultaneously. This comprehensive detection capability enhances the system's effectiveness in identifying tampered videos, along with types of tampering. The approach in Ref. [20] selected for comparison, utilizing CNN, demonstrates the highest detection accuracy among the state-of-the-art methods presented in Section 2. However, it is important to mention that this accuracy is reported on a proprietary dataset developed by the authors, which is not publicly accessible. Table 6 shows the computation of various comparison parameters—precision, recall, F1 score, and detection accuracy—of the proposed and state-of-the-art methods regarding CSVTED. When compared to certain existing methods (referred to as Ref. [20]), which require tampered frames to be in multiples of 10 and cannot detect tampering involving fewer than 25 frames, the proposed technique exhibits higher sensitivity, and it can detect tampering involving smaller numbers of frames, thus enhancing its effectiveness in detecting subtle manipulations. Furthermore, the method in Ref. [21] does not show robustness, as it has low accuracy in regards to our challenging CSVTED, and its localization is not precise.

Figure 12 shows the performance rates of TPR and DA of the proposed systems and state-of-the-art regarding CSVTED. The experimental results indicate that our deep learning approaches (CNN-GRU and CNN-LSTM) are capable of capturing various hidden features from STP images using VGG16, leading to high accuracies for detecting inter-frame tampering. VGG16 has been trained on a large dataset, like ImageNet, which contains a vast number of diverse images. As a result, the convolutional layers of VGG16 have learned rich representations of various visual patterns and objects. Leveraging these pre-learned features can be highly beneficial, especially when working with limited data or resources. Proposed Model-02 (2D-CNN with GRU-L1) achieves the highest accuracy of 90.73%, while using a small feature size of 512 per STP image, as compared to state-of-the-art

4096, which makes the proposed approach computationally efficient. The second proposed model, Model-06, demonstrates a 90.67% accuracy, despite having a significantly smaller feature size, only 3% of the size of the state-of-the-art model. This not only underscores the effectiveness of employing an autoencoder to reduce feature space, but also contributes to improving the extraction of discriminative features for the task of detecting video tampering, as depicted in Figures 4g–i and 4m–o.

Table 6. Detailed results of our 2D-CNN-based network with LSTM, GRU, and different feature sizes regarding CSVTED.

Method	Feature Size/STP Image	Forgeries Identified	Precision	Recall	F1 Score	Detection Accuracy
Fadl et al., 2021 [21]	4096	Insertion Deletion	0.738 0.719	0.0.91 0.603	0.813 0.655	73.45%
Proposed Model-01 (2D-CNN with LSTM-L1)	512	Insertion Deletion	0.9792 0.8529	0.9851 0.9336	0.9821 0.8904	90.53%
Proposed Model-02 (2D-CNN with GRU-L1)	512	Insertion Deletion	0.9877 0.8459	**0.9899** 0.9420	0.9887 0.8905	**90.73%**
Proposed Model-03 (2D-CNN with GRU-L2)	512	Insertion Deletion	0.9931 0.8432	0.9850 **0.9422**	**0.9889** 0.8890	90.55%
Proposed Model-04 (2D-CNN with GRU-L3)	512	Insertion Deletion	**0.9935** 0.8472	0.9831 **0.9422**	0.9882 0.8915	90.65
Proposed Model-05 (2D-CNN with GRU-L4)	512	Insertion Deletion	0.9891 0.8537	0.9854 0.9237	0.9872 0.8865	90.34
Proposed Model-06 (2D-CNN with GRU-L2)	128	Insertion Deletion	0.9761 **0.8703**	0.9835 0.9208	0.9797 **0.8943**	90.67

Highest values are represented in bold corresponding to each tampering.

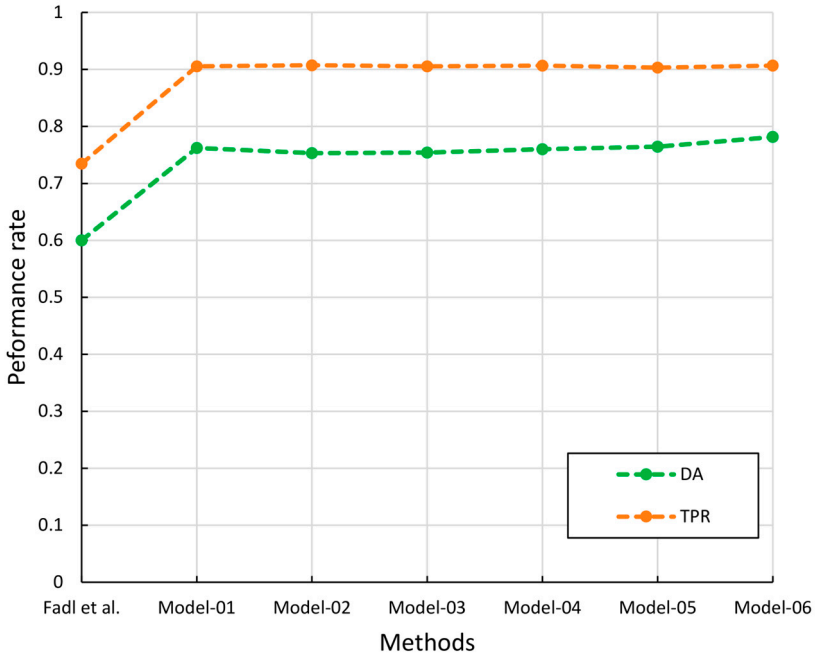


Figure 12. Performance rates of the proposed systems and state-of-the-art model [21] regarding CSVTED.

Figure 13 compares the accuracy of detecting and identifying tampering types at the video level. Kumar and Gaur [56] reported a detection accuracy of 47% for frame deletion and 77% for frame insertion tampering. They stated that detecting tampering

becomes highly complex when only a few frames are deleted from the sequence. The close relationship between the non-deleted frames at these locations contributes to this complexity, resulting in lower accuracy for detecting frame deletion forgery. In contrast, our Model-01 exhibits detection accuracy of 93% and 98.5% for frame deletion and insertion tampering, respectively, while Model-02 achieves detection accuracy of 94% for frame deletion and 99% for frame insertion tampering, even when dealing with a small number of tampered frames, which indicates that our method can efficiently detect both insertion and deletion tampering with high accuracy, without imposing strict constraints on the number of tampered frames; this represents a significant improvement over the accuracies of existing techniques in the field. Thus, LSTM-L1 and GRU-L1, with dimensions of 512, and GRU-L2, with dimensions of 128, outperform traditional methods and are recommended to detect multiple inter-frame tampering in videos.

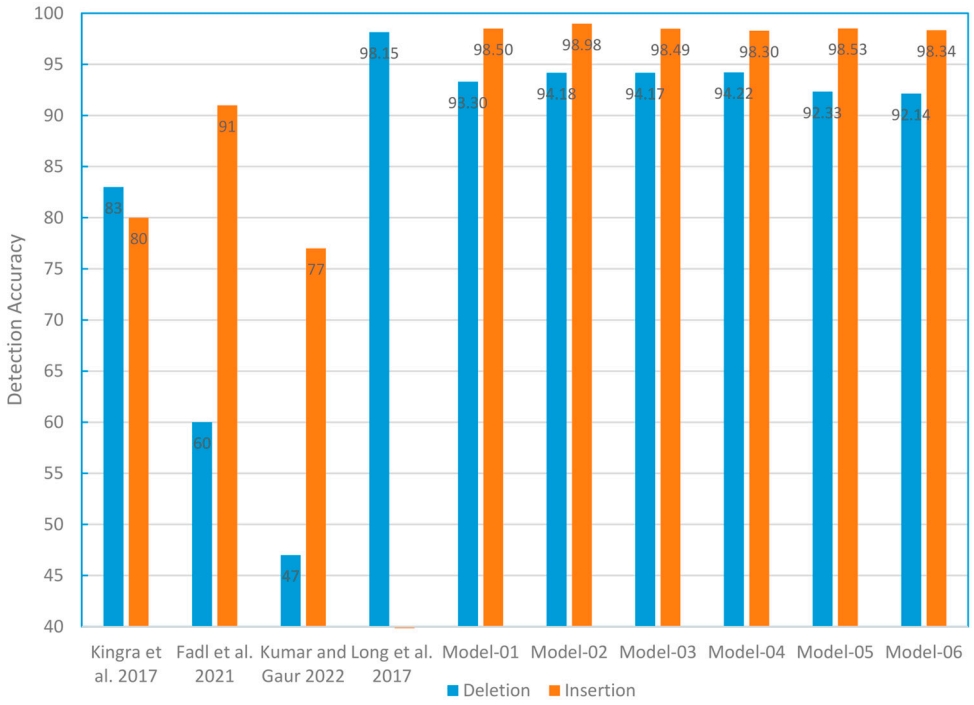


Figure 13. Comparison with the state-of-the-art methods for inter-frame tampering detection, along with type, at the video level [21,29,43,56].

The following are some advantages and limitations of the proposed system: First, the proposed system considers multiple inter-frame tampering in one 2D-CNN model, where each STP image of the video is processed independently; this avoids the need for extensive computational space and complex computations involved in processing the entire video sequences simultaneously, as required by 3D-CNNs. Second, the spatiotemporal feature fusion, coupled with LSTM/GRU to examine long-range dependencies among video frames, enhances the robustness of the detection process. Third, the feature dimensions are excessively high, leading to computational challenges. To address this, we suggest employing an autoencoder to diminish the feature space's dimensionality. This approach notably alleviates the computational burden of our method by reducing the dimensionality of feature space, while maintaining high detection accuracy. Last, our method used spatiotemporal deep automatic features, which achieved superior results in detecting, localizing, and determining the type of inter-frame tampering when compared to the state-of-the-art methods. It does not impose any constraints on frame rate, video formats, the type of capturing device, or the number of tampered frames; it is capable of detecting tampering involving as few as ten frames. However, there are some limitations: First, since our focus is on surveillance videos, the proposed system is unable to detect tampering if there is a scene change in the

video. Second, it is observed that by increasing the depth of LSTM/GRU, there is a slight decrease in performance. This may be due to model overfitting because additional layers increase the complexity of the model, making it harder to train and optimize it effectively, limiting our method to the use of simple model architecture. For complex models, more training data is required. Last, dimensionality reduction techniques inherently involve reducing the dimensionality of the feature space, which may result in a loss of information. It is important to balance dimensionality reduction with the preservation of discriminative information relevant to tampering detection.

Overall, the proposed method offers a robust approach to video tampering detection by leveraging CNNs for feature extraction and considering inter-frame dependencies. Its ability to detect, localize, and determine the type of multiple inter-frame tampering, without imposing strict constraints on the number of tampered frames, leads to superior performance compared to that of state-of-the-art methods.

To estimate computational cost, we monitor the models in terms of running time, as shown in Figure 14. Among these, both the LSTM and GRU models exhibit the shortest running time in terms of hours when utilizing a feature dimension of 128, and they are superior to the model with higher dimensional features. However, regarding detection accuracy, superior results are observed when employing features with 512 and 128 dimensions. Consequently, we do not compare the accuracy achieved with 256 dimensional features with these models. In the case of the LSTM model, the utilization of 512 dimensional features results in a runtime exceeding 9 h, whereas employing features with 128 dimensions requires approximately 5.5 h. Notably, the detection accuracy values with the GRU model closely align with those of the LSTM model, while requiring a shorter runtime. Specifically, the GRU model with 128 dimensional features consumed the shortest runtime of 1.5 h.

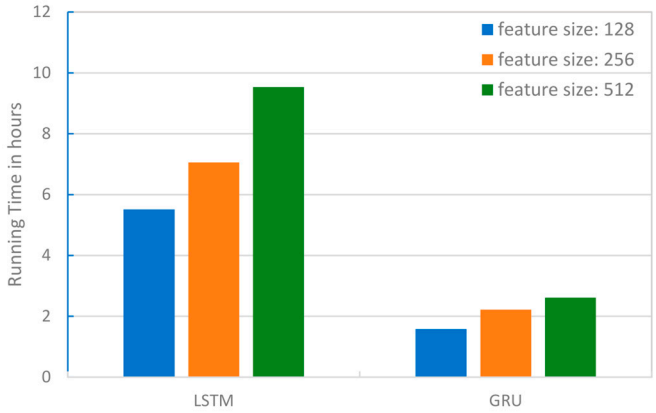


Figure 14. Running time of two models (in hours) with different feature dimensions.

This underscores the effectiveness of autoencoders in reducing data dimensionality, while preserving tampering information. Such dimensionality reduction, coupled with the preservation of key features, can substantially alleviate computational complexity, making autoencoders valuable tools for various tampering detection tasks based on deep learning.

6. Conclusions

Inter-frame tampering in surveillance videos undermines the integrity of video evidence, potentially influencing law enforcement agencies and court decisions. This paper proposes an innovative inter-frame tampering detection system that exploits deep feature fusion techniques. The system utilizes a 2D-CNN to extract features from the spatiotemporal average pooling of overlapped video clips. Overlapped clips preserve the temporal coherence in a better manner. To address computational challenges arising from the high dimensionality of the features, an autoencoder is employed. In our work, we show that deep features with reduced dimensionality are effective for video forensics, achieving high accuracy compared to high-dimensional feature approaches, which is beneficial for

real-world applications where video data is collected continuously and in large volumes. Long-range dependencies between distant frames are also captured by LSTM/GRU. The proposed system undergoes testing on our large dataset comprising 2555 videos, which includes challenging videos with different complexity levels, i.e., from a simple background to complex backgrounds, a single object to multiple objects, with random movement, and diverse lighting conditions such as morning, noon, evening, night, and fog. Notably, our technique does not restrict the minimum number of frames required to be inserted or deleted. Compared with handcrafted and deep feature-based methods, our approach offers a distinct advantage in detecting, localizing, and determining the type of inter-frame tampering (i.e., frame insertion and deletion), achieving over 90% accuracy, even with as few as ten tampered frames. By thorough comparison of simple LSTM and GRU models using a statistical paired *t*-test, we recommend single/two-layer LSTM and GRU models with appropriate parameters for inter-frame tampering detection. The experimental results validate the efficacy of the proposed method in detecting, localizing, and determining the tampering type in videos, demonstrating superiority over the state-of-the-art techniques. In the future, the frame duplication type of tampering can also be considered. Additionally, we intend to enhance the system's capability to detect multiple inter-frame tampering in a single video, thus fortifying its utility in real-world applications.

Author Contributions: Conceptualization, M.H. and Z.H.; data curation and formal analysis, N.A.; investigation, N.A.; methodology, N.A.; supervision, M.H. and Z.H.; writing—original draft, N.A. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the Researchers Supporting Project, number (RSP2024R109), King Saud University, Riyadh, Saudi Arabia.

Data Availability Statement: The datasets generated during and/or analyzed during the current study are available from the corresponding author upon reasonable request.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Lu, C.S.; Liao, H.Y.M. Structural digital signature for image authentication: An incidental distortion resistant scheme. In Proceedings of the 2000 ACM Workshops on Multimedia, Los Angeles, CA, USA, 30 October–3 November 2000.
2. Tolosana, R.; Vera-Rodriguez, R.; Fierrez, J.; Ortega-Garcia, J. Feature-based dynamic signature verification under forensic scenarios. In Proceedings of the 3rd International Workshop on Biometrics and Forensics (IWBF 2015), Gjøvik, Norway, 3–4 March 2015.
3. Pawar, A.; Sheikh, H.; Dhawade, N. Data encryption and security using video watermarking. *Int. J. Eng. Sci.* **2016**, *4*, 3238.
4. Sitara, K.; Mehtre, B. Digital video tampering detection: An overview of passive techniques. *Digit. Investig.* **2016**, *18*, 8–22. [CrossRef]
5. Singh, R.D.; Aggarwal, N. Video content authentication techniques: A comprehensive survey. *Multimed. Syst.* **2018**, *24*, 211–240. [CrossRef]
6. Shelke, N.A.; Kasana, S.S. Multiple forgeries identification in digital video based on correlation consistency between entropy coded frames. *Multimed. Syst.* **2022**, *28*, 267–280. [CrossRef]
7. Akhtar, N.; Saddique, M.; Asghar, K.; Bajwa, U.I.; Hussain, M.; Habib, Z. Digital Video Tampering Detection and Localization: Review, Representations, Challenges and Algorithm. *Mathematics* **2022**, *10*, 168. [CrossRef]
8. Long, C.; Basharat, A.; Hoogs, A.; Singh, P.; Farid, H. A Coarse-to-fine Deep Convolutional Neural Network Framework for Frame Duplication Detection and Localization in Forged Videos. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops. 2019, Long Beach, CA, USA, 16–20 June 2019.
9. Johnston, P.; Elyan, E. A review of digital video tampering: From simple editing to full synthesis. *Digit. Investig.* **2019**, *29*, 67–81. [CrossRef]
10. Wang, W.; Farid, H. Exposing digital forgeries in video by detecting duplication. In Proceedings of the 9th Workshop on Multimedia & Security 2007, Dallas, TX, USA, 20–21 September 2007.
11. Wang, Q.; Li, Z.; Zhang, Z.; Ma, Q. Video Inter-Frame Forgery Identification Based on Consistency of Correlation Coefficients of Gray Values. *J. Comput. Commun.* **2014**, *2*, 51–57. [CrossRef]
12. Singh, G.; Singh, K. Video frame and region duplication forgery detection based on correlation coefficient and coefficient of variation. *Multimed. Tools Appl.* **2019**, *78*, 11527–11562. [CrossRef]
13. Zhang, Z.; Hou, J.; Ma, Q.; Li, Z. Efficient video frame insertion and deletion detection based on inconsistency of correlations between local binary pattern coded frames. *Secur. Commun. Netw.* **2015**, *8*, 311–320. [CrossRef]

14. Fadl, S.; Megahed, A.; Han, Q.; Qiong, L. Frame duplication and shuffling forgery detection technique in surveillance videos based on temporal average and gray level co-occurrence matrix. *Multimed. Tools Appl.* **2020**, *79*, 17619–17643. [CrossRef]
15. Kharat, J.; Chougule, S. A passive blind forgery detection technique to identify frame duplication attack. *Multimed. Tools Appl.* **2020**, *79*, 8107–8123. [CrossRef]
16. Feng, C.; Xu, Z.; Jia, S.; Zhang, W.; Xu, Y. Motion-adaptive frame deletion detection for digital video forensics. *IEEE Trans. Circuits Syst. Video Technol.* **2016**, *27*, 2543–2554. [CrossRef]
17. Jia, S.; Xu, Z.; Wang, H.; Feng, C.; Wang, T. Coarse-to-Fine Copy-Move Forgery Detection for Video Forensics. *IEEE Access* **2018**, *6*, 25323–25335. [CrossRef]
18. Nguyen, X.H.; Hu, Y.; Amin, M.A.; Hayat, K.G.; Le, V.T.; Truong, D.-T. Detecting video inter-frame forgeries based on convolutional neural network model. *Int. J. Image, Graph. Signal Process.* **2020**, *3*, 1–12.
19. Zampoglou, M.; Markatopoulou, F.; Mercier, G.; Touska, D.; Apostolidis, E.; Papadopoulos, S.; Cozien, R.; Patras, I.; Mezaris, V.; Kompatsiaris, I. Detecting Tampered Videos with Multimedia Forensics and Deep Learning. In Proceedings of the International Conference on Multimedia Modeling 2019, Thessaloniki, Greece, 8–11 January 2019; Springer: Berlin/Heidelberg, Germany, 2019.
20. Johnston, P.; Elyan, E.; Jayne, C. Video tampering localisation using features learned from authentic content. *Neural Comput. Appl.* **2020**, *32*, 12243–12257. [CrossRef]
21. Fadl, S.; Han, Q.; Li, Q. CNN spatiotemporal features and fusion for surveillance video forgery detection. *Signal Process. Image Commun.* **2021**, *90*, 116066. [CrossRef]
22. Yu, L.; Wang, H.; Han, Q.; Niu, X.; Yiu, S.; Fang, J.; Wang, Z. Exposing frame deletion by detecting abrupt changes in video streams. *Neurocomputing* **2016**, *205*, 84–91. [CrossRef]
23. Ulutas, G.; Ustubioglu, B.; Ulutas, M.; Nabiyev, V.V. Frame duplication detection based on bow model. *Multimed. Syst.* **2018**, *24*, 549–567. [CrossRef]
24. Bozkurt, I.; Ulutas, G. Detection and localization of frame duplication using binary image template. *Multimed. Tools Appl.* **2023**, *82*, 31001–31034. [CrossRef]
25. Alsakar, Y.M.; Mekky, N.E.; Hikal, N.A. Detecting and Locating Passive Video Forgery Based on Low Computational Complexity Third-Order Tensor Representation. *J. Imaging* **2021**, *7*, 47. [CrossRef]
26. Sitara, K.; Mehtre, B. A comprehensive approach for exposing inter-frame video forgeries. In Proceedings of the 2017 IEEE 13th International Colloquium on Signal Processing & Its Applications (CSPA), Penang, Malaysia, 10–12 March 2017.
27. Bakas, J.; Naskar, R. A Digital Forensic Technique for Inter–Frame Video Forgery Detection Based on 3D CNN. In Proceedings of the International Conference on Information Systems Security 2018, Funchal, Purtugal, 22–24 January 2018; Springer: Berlin/Heidelberg, Germany, 2018.
28. Tyagi, S.; Yadav, D. A detailed analysis of image and video forgery detection techniques. *Vis. Comput.* **2023**, *39*, 813–833. [CrossRef]
29. Long, C.; Smith, E.; Basharat, A.; Hoogs, A. A c3d-based convolutional neural network for frame dropping detection in a single video shot. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Honolulu, HI, USA, 21–26 July 2017.
30. Subramanyam, A.V.; Emmanuel, S. Pixel estimation based video forgery detection. In Proceedings of the 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, Vancouver, BC, Canada, 26–31 May 2013.
31. Qureshi, M.A.; Deriche, M. A bibliography of pixel-based blind image forgery detection techniques. *Signal Process. Image Commun.* **2015**, *39*, 46–74. [CrossRef]
32. Fayyaz, M.A.; Anjum, A.; Ziauddin, S.; Khan, A.; Sarfaraz, A. An improved surveillance video forgery detection technique using sensor pattern noise and correlation of noise residues. *Multimed. Tools Appl.* **2020**, *79*, 5767–5788. [CrossRef]
33. Kaur, H.; Jindal, N. Deep convolutional neural network for graphics forgery detection in video. *Wirel. Pers. Commun.* **2020**, *112*, 1763–1781. [CrossRef]
34. Lin, G.-S.; Chang, J.-F.; Chuang, C.-H. Detecting frame duplication based on spatial and temporal analyses. In Proceedings of the 2011 6th International Conference on Computer Science & Education (ICCSE), Singapore, 3–5 August 2011.
35. Kumari, P.; Kaur, M. Empirical evaluation of motion Cue for passive-blind video tamper detection using optical flow technique. In Proceedings of the International Joint Conference on Advances in Computational Intelligence: IJCACI 2021, Dhaka, Bangladesh, 23–24 October 2021; Springer: Berlin/Heidelberg, Germany, 2022.
36. Chen, S.; Tan, S.; Li, B.; Huang, J. Automatic detection of object-based forgery in advanced video. *IEEE Trans. Circuits Syst. Video Technol.* **2015**, *26*, 2138–2151. [CrossRef]
37. Tan, S.; Chen, S.; Li, B. GOP based automatic detection of object-based forgery in advanced video. In Proceedings of the 2015 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA), Hong Kong, China, 16–19 December 2015.
38. Voronin, V.V.; Sizyakin, R.; Svirin, I.; Zelensky, A.; Nadykto, A. Detection of deleted frames on videos using a 3D convolutional neural network. In Proceedings of the Counterterrorism, Crime Fighting, Forensics, and Surveillance Technologies II, Berlin, Germany, 10–11 September 2018.
39. Al-Sanjary, O.I.; Sulong, G. Detection of Video Forgery: A Review of Literature. *J. Theor. Appl. Inf. Technol.* **2015**, *74*, 208–220.
40. Fadl, S.M.; Han, Q.; Li, Q. Authentication of surveillance videos: Detecting frame duplication based on residual frame. *J. Forensic Sci.* **2018**, *63*, 1099–1109. [CrossRef]
41. Sitara, K.; Mehtre, B. Detection of inter-frame forgeries in digital videos. *Forensic Sci. Int.* **2018**, *289*, 186–206. [CrossRef]

42. Wang, Q.; Li, Z.; Zhang, Z.; Ma, Q. Video inter-frame forgery identification based on optical flow consistency. *Sens. Transducers* **2014**, *166*, 229.
43. Kingra, S.; Aggarwal, N.; Singh, R.D. Inter-frame forgery detection in H. 264 videos using motion and brightness gradients. *Multimed. Tools Appl.* **2017**, *76*, 25767–25786. [CrossRef]
44. Singh, R.D.; Aggarwal, N. Optical flow and prediction residual based hybrid forensic system for inter-frame tampering detection. *J. Circuits Syst. Comput.* **2017**, *26*, 1750107. [CrossRef]
45. Stamm, M.C.; Lin, W.S.; Liu, K.J.R. Temporal forensics and anti-forensics for motion compensated video. *IEEE Trans. Inf. Forensics Secur.* **2012**, *7*, 1315–1329. [CrossRef]
46. Bakas, J.; Naskar, R.; Bakshi, S. Detection and localization of inter-frame forgeries in videos based on macroblock variation and motion vector analysis. *Comput. Electr. Eng.* **2021**, *89*, 106929. [CrossRef]
47. Huang, C.C.; Lee, C.E.; Thing, V.L.L. A Novel Video Forgery Detection Model Based on Triangular Polarity Feature Classification. *Int. J. Digit. Crime Forensics* **2020**, *12*, 14–34. [CrossRef]
48. Shanableh, T. Detection of frame deletion for digital video forensics. *Digit. Investig.* **2013**, *10*, 350–360. [CrossRef]
49. Chao, J.; Jiang, X.; Sun, T. A novel video inter-frame forgery model detection scheme based on optical flow consistency. In Proceedings of the International Workshop on Digital Watermarking 2012, Shanghai, China, 31 October–3 November 2012; Springer: Berlin/Heidelberg, Germany, 2012.
50. Feng, C.; Xu, Z.; Zhang, W.; Xu, Y. Automatic location of frame deletion point for digital video forensics. In Proceedings of the 2nd ACM Workshop on Information Hiding and Multimedia Security 2014, Salzburg, Austria, 11–13 June 2014.
51. Liao, S.Y.; Huang, T.Q. Video copy-move forgery detection and localization based on Tamura texture features. In Proceedings of the 6th International Congress on Image and Signal Processing (CISP) 2013, Hangzhou, China, 16–18 December 2013.
52. Zhao, D.-N.; Wang, R.-K.; Lu, Z.-M. Inter-frame passive-blind forgery detection for video shot based on similarity analysis. *Multimed. Tools Appl.* **2018**, *77*, 25389–25408. [CrossRef]
53. Bakas, J.; Naskar, R.; Dixit, R. Detection and localization of inter-frame video forgeries based on inconsistency in correlation distribution between Haralick coded frames. *Multimed. Tools Appl.* **2019**, *78*, 4905–4935. [CrossRef]
54. Shehnaz; Kaur, M. Detection and localization of multiple inter-frame forgeries in digital videos. *Multimed. Tools Appl.* **2024**, 1–33. [CrossRef]
55. Cheng, X.; Zhang, M.; Lin, S.; Li, Y.; Wang, H. Deep Self-Representation Learning Framework for Hyperspectral Anomaly Detection. *IEEE Trans. Instrum. Meas.* **2023**, *73*, 1–16. [CrossRef]
56. Kumar, V.; Gaur, M. Multiple forgery detection in video using inter-frame correlation distance with dual-threshold. *Multimed. Tools Appl.* **2022**, *81*, 43979–43998. [CrossRef]
57. Huang, T.; Zhang, X.; Huang, W.; Lin, L.; Su, W. A multi-channel approach through fusion of audio for detecting video inter-frame forgery. *Comput. Secur.* **2018**, *77*, 412–426. [CrossRef]
58. Panchal, H.D.; Shah, H.B. Multiple forgery detection in digital video based on inconsistency in video quality assessment attributes. *Multimedia Syst.* **2023**, *29*, 2439–2454. [CrossRef]
59. Nixon, M.; Aguado, A. *Feature Extraction and Image Processing for Computer Vision*, 3rd ed.; Academic Press: Oxford, UK, 2012. [CrossRef]
60. Han, D. Comparison of commonly used image interpolation methods. In Proceedings of the 2nd International Conference on Computer Science and Electronics Engineering (ICCSEE 2013), Hangzhou, China, 22–23 March 2013; Atlantis Press: Dordrecht, The Netherlands, 2013.
61. Patil, M. Interpolation techniques in image resampling. *Int. J. Eng. Technol.* **2018**, *7*, 567–570.
62. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. *Adv. Neural Inf. Process. Syst.* **2012**, *1*, 1097–1105. [CrossRef]
63. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition 2014, Columbus, OH, USA, 23–28 June 2014.
64. Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, USA, 20–25 June 2009.
65. Weiss, K.; Khoshgoftaar, T.M.; Wang, D.D. A survey of transfer learning. *J. Big Data* **2016**, *3*, 1345–1459. [CrossRef]
66. Zhuang, F.; Qi, Z.; Duan, K.; Xi, D.; Zhu, Y.; Zhu, H.; Xiong, H.; He, Q. A comprehensive survey on transfer learning. *Proc. IEEE* **2020**, *109*, 43–76. [CrossRef]
67. Sakurada, M.; Yairi, T. Anomaly detection using autoencoders with nonlinear dimensionality reduction. In Proceedings of the MLSDA 2014 2nd Workshop on Machine Learning for Sensory Data Analysis, Gold Coast, QLD, Australia, 2 December 2014.
68. Pinaya, W.H.L.; Vieira, S.; Garcia-Dias, R.; Mechelli, A. *Autoencoders, in Machine Learning*; Elsevier: Amsterdam, The Netherlands, 2020; pp. 193–208.
69. Bank, D.; Koenigstein, N.; Giryes, R. Autoencoders. In *Machine Learning for Data Science Handbook: Data Mining and Knowledge Discovery Handbook*; Springer: Berlin/Heidelberg, Germany, 2023; pp. 353–374.
70. Bengio, Y.; Lamblin, P.; Popovici, D.; Larochelle, H. Greedy layer-wise training of deep networks. *Adv. Neural Inf. Process. Syst.* **2006**, *19*, 153–160.
71. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef]

72. Chung, J.; Gulcehre, C.; Cho, K.; Bengio, Y. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv* **2014**, arXiv:1412.3555.
73. Shewalkar, A.; Nyavanandi, D.; Ludwig, S.A. Performance evaluation of deep neural networks applied to speech recognition: RNN, LSTM and GRU. *J. Artif. Intell. Soft Comput. Res.* **2019**, *9*, 235–245. [CrossRef]
74. Fadl, S.; Han, Q.; Qiong, L. Exposing video inter-frame forgery via histogram of oriented gradients and motion energy image. *Multidimens. Syst. Signal Process.* **2020**, *31*, 1365–1384. [CrossRef]
75. Ulutas, G.; Ustubioglu, B.; Ulutas, M.; Nabiyev, V. Frame duplication/mirroring detection method with binary features. *IET Image Process.* **2017**, *11*, 333–342. [CrossRef]
76. Brunet, D.; Vrscay, E.R.; Wang, Z. On the mathematical properties of the structural similarity index. *IEEE Trans. Image Process.* **2011**, *21*, 1488–1499. [CrossRef]
77. Yoo, C.D.; Shi, Y.-Q.; Kim, H.J.; Piva, A.; Kim, G. *Digital Forensics and Watermarking: 17th International Workshop, IWDW 2018, Jeju Island, Republic of Korea, 22–24 October 2018*; Springer: Berlin/Heidelberg, Germany, 2019; Proceedings Volume 11378.