Detecting Frame Deletion in Videos Using Supervised and Unsupervised Learning with Convolutional Neural Networks

Jorge Ceron, Cristian Tinipuclla, and Pedro Shiguihara, *Member, IEEE*

***Abstract*—In recent years videos are susceptible not only to any edition, but also to a variety of forgeries. One of the most popular video forgeries is frame deletion, in which a group of frames is removed to hide specific actions from the human eye. When frame deletion occurs, videos selected as evidence lose their evidentiary value. This highlights the necessity of automation, especially for analyzing large volumes of videos. Thus, the performance of two deep learning approaches for frame deletion detection is measured through k-fold cross validation in three datasets and its combinations. Both of the approaches use Convolutional Neural Networks (CNN): The first one, a supervised 3DCNN model and, the second one, is an unsupervised model compound of VGG-16. Based on this, the main contributions of this study are: the release of code for a novel comparison between two approaches using 5-fold and 10-fold cross-validation; the introduction of a new dataset for the research community, called the Driving Test Dataset (DTD); and the development of a more realistic forgery process for the datasets, which is explained in detail. One of the key findings is that the highest performance was achieved using the first-mentioned model on the combined VIFFD and DTD datasets, reaching a precision of 0.94, recall of 0.93, F1-score of 0.92, and an accuracy of 0.93. Finally, the results are analyzed and recommendations are provided to guide future work in this area.**

Link to graphical and video abstracts, and to code:
https://latamt.ieeer9.org/index.php/transactions/article/view/9568

***Index Terms*—frame deletion detection, deep learning, CNN, video forgery detection, temporal forgery**

I. Introduction

With the advent of modern technology, videos have become increasingly susceptible to forgery, which can distort reality and influence decisions detrimentally [1], [2], and cause defamation, frauds, misdirection and even harm to individuals or organizations [3]. Advances in both simple applications like Tiktok and sophisticated editing tools such as Adobe Premier Pro [3], [4], have made video alterations more accessible. There exists a wide range of video forgery, and can occur at either the spatial level, or temporal level. Spatial level forgeries can refer to copy-move, splicing, inpainting and illumination changes, which can also be found on image forgeries. In contrast, temporal-level forgeries are intrinsic to video content due to its fundamental characteristics and refer to inter-frame manipulations [5], which consists of any form of altering the continuity of frames in a video, such as insertion, frame shuffling, frame duplication, and frame deletion [6], with frame deletion being one of the significant issues to which videos are susceptible [2]. Frame deletion involves the removal of specific frames from a video sequence, potentially obscuring key scenes [1], [7]. This issue is particularly critical when surveillance videos, whose integrity has not been verified, are presented as evidence in court. Consequently, detecting video forgery, and in particular frame deletion, is essential to validating whether video content has undergone intentional manipulation [8].

Frame deletion detection is mainly addressed in the literature through passive approaches, which rely on identifying inherent attributes and patterns in videos to predict whether they have been forged [9]. In this context, various techniques for detecting frame deletion have been proposed, broadly categorized into standard and deep learning-based methods [10]. The standard method workflow involves analyzing video features such as optical flow, macroblock, motion or time-frequency analysis. However, this workflow lacks computational efficiency, as its processing tasks are time-consuming [10]. In contrast, the deep learning-based workflow uses Artificial Neural Networks (ANNs) to automate the feature extraction and classification stages of video. Among these, Convolutional Neural Networks (CNNs) are the preferred choice due to their exceptional ability to process high-dimensional features and provide the required information efficiently [11]. CNN is compound of three main layers: first, convolutional; second, pooling and finally fully connected [12]. This type of ANN has demonstrated along various academic researches its potential to effectively address the task of retrieving dimensional features from images [11].

CNNs for detecting frame deletion can be utilized in both supervised and unsupervised learning approaches. In the supervised approach, the use of pretrained CNNs from the literature [13], [14] is common, or their architectures are adapted to create new models focused on frame deletion tasks [10], [14]–[18]. VGG-16 network, pretrained in ImageNet Dataset [19](a dataset with more than a million of images) has been widely employed to detect manipulation in videos, primarily because of its small 3X3 kernels, proven to have high precision in detecting subtle signs of tampering, such as video frame removal [20]. In the unsupervised approach, CNNs are typically used for feature extraction, followed by statistical operations to detect or classify the forgery [1], [21], [22]. In contrast to conventional CNNs, a 3D-CNN is specially

The associate editor coordinating the review of this manuscript and approving it for publication was Eduardo José da Luz (*Corresponding author: Pedro Shiguihara*).

J. Ceron, C. Tinipuclla, and Pedro Shiguihara are with the Universidad San Ignacio de Loyola, Lima, Perú (e-mails: jorge.ceron@usil.pe, cristian.tinipuclla@usil.pe, and pshiguihara@usil.edu.pe).

designed to handle videos since its ability to learn and retrieve spatio-temporal features [23]. It has been demonstrated the superiority of CNNs over other machine learning methods in forgery detection tasks. However, there is a clear downside, the fact this networks require high computational power [24]. Unlike CNNs, statistical techniques such as Pearson's correlation coefficient is used to identify points of any type of temporal forgery [25]. Moreover, frame difference is effective at distinguishing frame deletion since determines variations in the correlation factor of two sequential frames [18].

In this work, we present a novel comparison between supervised and unsupervised deep learning approaches—based on models from [18] and [21]—using both 5-fold and 10-fold cross-validation. Performance was assessed using accuracy, F1-score, precision, and recall on two public datasets, VIFFD [26] and UCF-101 [27], and a new dataset introduced in this study: the Driving Test Dataset (DTD) [28]. By combining these datasets into seven test sets, the evaluation was made more robust. Building on the previous work [28], this research introduces methodological enhancements such as improved architecture, a broader literature review, the application of k-fold cross-validation, and notable improvements in key performance metrics, especially for the unsupervised approach.

Thus, the main contributions of this study are as follows:

1) The release of source code on GitHub for a novel comparison between two approaches using both 5-fold and 10-fold cross-validation.
2) The introduction of a new dataset for the research community, called the Driving Test Dataset (DTD), consisting of over 100 driving test videos released under a Creative Commons license.
3) The development of a more realistic forgery process for the datasets, in contrast to prior methods used in [18] and [21], which involved only deleting a fixed number of frames.
4) A comprehensive evaluation of the performance of two deep learning approaches for frame deletion detection, conducted using both 5-fold and 10-fold cross-validation schemes.

This paper is organized as follows: Section II reviews related work on frame deletion. Section III provides a detailed description of the proposal. Section IV outlines the experimental configuration. Section V presents the results. Section VI discusses the results reported in the previous section. Finally, Section VII, concludes the paper and offer recommendations for future research.

II. Related Work

This section reviews standard methods and deep learning-based approaches to detect frame deletion. In standard methods, [29] defines a system that detects abrupt changes in videos using prediction residual and the number of intra macroblocks but fails with slow-motion videos, achieving no more than 88% accuracy. Similarly [30], utilizes motion fluctuation features to adapt to varying motion strengths, obtaining a 90% true positive rate for forged location detection but shares the same limitation. Then, in [31], a framework using Triangular Polarity Feature Classification (TPFC) achieved up to 89.76% precision rate in detecting frame deletion. However, diverse training conditions are needed to enhance feature discriminability. An alternative method was proposed in [32], based on Electric Network Frequency (ENF) phase anomalies but depends heavily on lighting, limiting its applicability. Optical flow orientation variations in [33] achieve an 89% detection rate across different coding types, though false alarms remain lower than 10% only with 5 moving objects. In [34] the proposal is based on analyzing compressed domain video footprints and motion vector. Experiments used varying constant rate factor (crf) = 18 and varying bit rate (BR), obtaining 90.47% accuracy, 95% recall and 92.68% F1-Score. In [25] a method based on the high-frequency features of reconstructed coefficients is defined, though it is tested only in videos with MPEG-2 codec.

On the other hand, there exist deep learning-based methods for frame deletion detection, such as Long et al. [16], who proposed a 3DCNN in conjuction with cue-based techniques, achieving 98.2% of accuracy, although fails in videos with static motion. In case of Hong et al. [15], they design a system with feature extraction and classification with a Multilayer Perceptron (MLP), obtaining 88% of accuracy as results. Singla et al. [13], apply transfer learning in CNN models such as GoogleNet, ResNet, DenseNet or InceptionV3. They obtain an accuracy of 99% but its complexity and such lots of parameters decrease the processing speed. In [1], a CNN and a Gaussian Radial Basis Function Support Vector Machine (RBF-MSVM) technique were used to get 98.7% of accuracy as final results. Singla et al. [17] compared CNN, SVM and MLP, achieving the best result with the CNN. However, they only tested with 25 frames or 100 frames deleted, which limits tampering use cases. In [18], a model based on a 3DCNN with a difference frame technique is proposed, having 99% of accuracy as results, but again, they only worked with limited frame deletion use cases. Then, in [21] a CNN based on VGG16 with an analytical technique such as Pearson's correlation coefficient (PCC) obtained a 91% of accuracy. Its network architecture lacks of specification though. Unlike typical CNN methods, Yi Gong et al. [14] innovate with a technique called "Improved Residual Feature" - IReF, a technique for better recognition of frame deletion points using 3DCNN and Resnet, in which outstanded the most. In [10] a 3DCNN with a difference layer is proposed not only to detect frame forgery, but also to localize it, implementing a Multi-scale structural similarity (MS-SSIM) index measurement and improving the accuracy from 82% to 98%. To address the problem of detecting video tampering on static camera videos, [22] presents an unsupervised approach focused on inter-frame local features calculation and an ANN to classify them. Finally, there is a post processing (rank filtering) and the final classification stage (with results of rank filtering). They experimented on two static camera videos (Road video and SU_corridor video), obtaining 88% accuracy on Road video and 89% on SU_corridor video.

In summary, this study replicates the approaches of [18] and [21] incorporating modifications on their architectures for improved evaluation of frame deletion detection.

III. Method

A. Comparison of Models

In this study, seven models were evaluated for frame deletion detection: **P1**, based on [18], and six variants of **P2**, which is derived from [21]. The six **P2** variants including its original architecture and five models pretrained on the ImageNet dataset: ResNet [35], DenseNet [36], InceptionV3, NasNet [37], and VGG-16 [38]. These models were chosen for their high accuracy in previous studies (up to 97.5% [13]) and are expected to perform well here. Unlike most existing methods, **P1** integrates a 3D CNN, while **P2** takes a different approach by leveraging CNNs exclusively for feature extraction, differing from typical end-to-end deep learning models.

Since **P2** uses CNNs exclusively for feature extraction, it allows easy integration of various pretrained models commonly used in video forgery detection, enabling performance comparison with the original model by [21]. VGG-16, used effectively in [1], [13] and also in [21], has shown strong performance in frame deletion detection. ResNet, adopted in [14], and DenseNet offer robust architectures with enhanced training speed and accuracy due to their multi-layer connections. Similarly, InceptionV3 and NasNet maintain consistent layer connections, helping manage computational costs. These advantages are also highlighted in [13]. Comparing these models with the original approach helped identify the most suitable architecture for this task.

The performance of the seven models was compared using 10-fold and 5-fold cross-validation on VIFFD, UCF-101, and DTD datasets. This approach ensures robust and reliable results by training on more diverse data subsets and improving generalization. Then, the evaluation of metrics employed statistical methods such as ROC curves and hypothesis testing. Algorithm 1 presents the steps followed in the proposed process. The GitHub repository[1] containing the code is publicly available.

Algorithm 1 Forgery Detection Evaluation Process

Input: VIFFD, UCF-101, and DTD datasets
Output: Performance evaluation results
Step 1: Conduct 10-fold cross-validation with P1 model
Save predicted labels for each fold
Step 2: Generate predictions using P2 model
Ensure consistent test sets
Step 3: Compile all predictions across folds
Step 4: Apply statistical analyses (e.g., ROC curves, hypothesis testing)
Step 5: Draw conclusions based on statistical results

The supervised architecture, referred to as (**P1**), is based on [18] and consists of the following stages (see Fig. 1): (1) input videos, (2) frame extraction, (3) preprocessing, (4) frame difference calculation, (5) training, and (6) output. The ConvLSTM2D layer was removed because the original paper does not explicitly specify its role within the architecture, and preliminary experiments showed that it did not yield satisfactory results. The process begins with (2), where frames are extracted from the input video and resized to 64×64 pixels in (3) to ensure consistency and computational efficiency. In (4), the difference between consecutive frames is computed to detect significant changes that may indicate manipulation, and these difference frames are then fed into (5), where a 3D Convolutional Neural Network (3DCNN) analyzes them to learn distinguishing patterns. Finally, (6) classifies the video as either original or forged based on the extracted features.

In contrast, (**P2**), derived from [21], consists of seven stages (see Fig. 2): (1) video input, (2) frame extraction, (3) preprocessing, (4) feature extraction, (5) PCC calculations, (6) adaptive thresholding, and (7) output. When developing the five alternative models based on **P2**, stages (2) and (3) were modified to accommodate pretrained networks, which require RGB channels. Specifically, in (5), lambda (λ) serves as the initial threshold for forgery detection: if the PCC difference is less than or equal to λ, the frame is classified as original; otherwise, it is marked as forged. In (6), the classification process incorporates Sigma ((δ)), which determines the upper (ub) and lower (lb) supporting values. These values, along with the threshold control parameters (γ_1) and (γ_1), define the adaptive thresholds that ultimately classify a video as original, deleted, or frame-inserted (though the latter was not implemented in this work). For further details, refer to Sections 4.2, 4.3, and 4.4 of Kumar et al. [21].

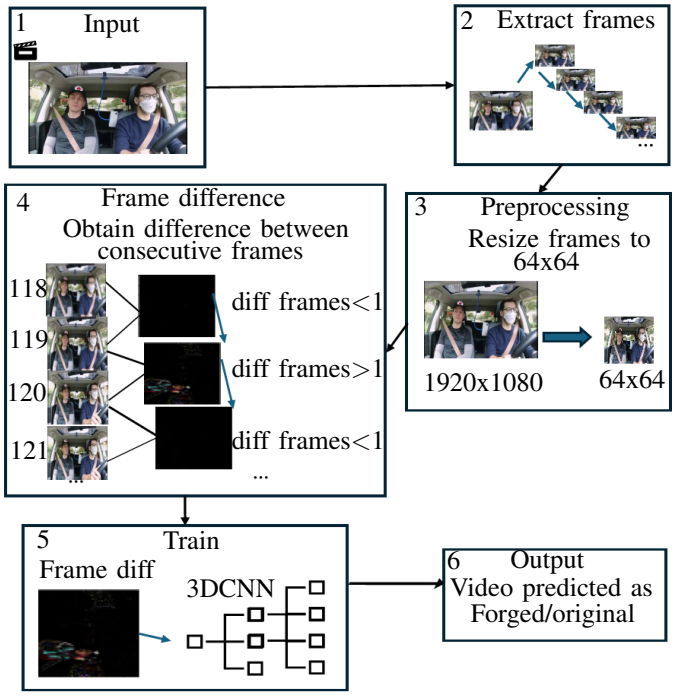


Fig. 1. The architecture of P1, inspired by [18], starts with (1) video input, (2) frame extraction, (3) preprocessing, where the video is resized to 64x64 pixels, (4) frame difference between consecutive frames, which are then used in the fifth block for (5) training a 3D CNN model. The final block, (6) the output which classifies the video as original or forged.

[1]https://github.com/1720907/GICC-VIDEOFRAUD-METHODS

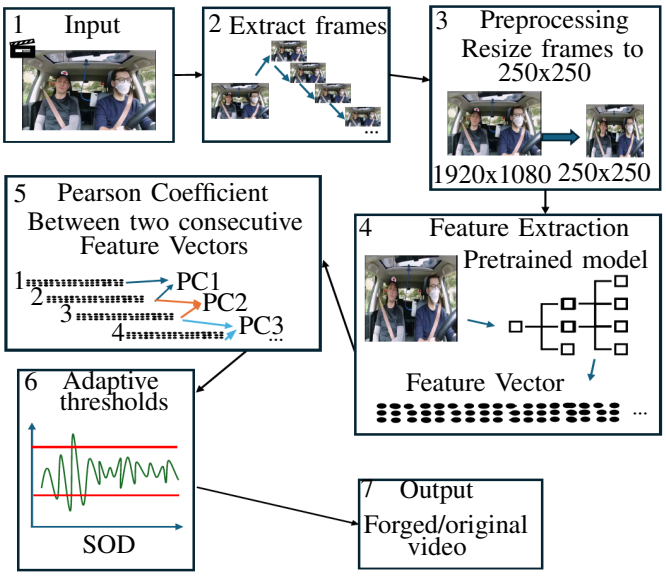


Fig. 2. The architecture of P2, inspired by [21], begins with (1) video input, (2) frame extraction, (3) preprocessing, in which the frames are resized to 250x250 pixels. Then, (4) feature extraction is applied using a pretrained model, (5) Pearson Correlation Coefficient (PCC) calculation between consecutive frames. (6) Adaptive thresholds and (7) output from classification

B. Creation of Datasets

In this study, different datasets were created by manipulating well-known databases: UCF-101 [27], VIFFD [26], and the DTD database [28], specifically using the "original" folder of unaltered videos. Details about the datasets can be found in [28], page 5, where the structure of the datasets and combined datasets is described. UCF-101 is denoted as D1, VIFFD as D2, and DTD as D3, while combined datasets are represented as D1+D2, D1+D3, D2+D3, and D1+D2+D3.

The Datasets were reorganized to improve video reading efficiency. The new structure includes:

- 'forgery_data': Manipulated videos.
- 'forgery_info': A CSV file with ground truth details, such as deletion points and manipulated frame indices.
- 'forgery_labels': .npy files with frame labels (0 for original, 1 for forged) used by the P2 model.
- 'p1_data_batches': Preprocessed frames in batches of 36 difference frames.
- 'p1_label_batches': Labels for each video matching the 36-frame batches.

This structure enhances the use of pandas DataFrames for efficient CSV reading, enabling quick access to file paths and names with sorting capability. It supports dataset generators (via TensorFlow's Dataset.from_generator), simplifying training, shuffling, and handling combined datasets.

C. Forgery Process

To identify manipulations, in this study methods from [21], [1], and [18] were adapted. As outlined in Equation 1 of [28], up to 10% of a video's frames are randomly deleted from a continuous section, with a minimum of 10 frames removed. Metadata, including video name, frame rate, indexes of deleted frames, total frame count, deletion time, and frame index, is saved in the "forgery_info" folder as a CSV file.

Fig. 3 shows a sequence from a forged video in the DTD dataset (*"deleted KATIE GETS HER DRIVERS PERMIT!! __first time driving_4_2"*). In this example, 59 frames were deleted between the 228th and 229th frames. The 229th frame, whose border is highlighted in red, serves as the ground truth since it directly follows the deletion. This visualization illustrates the labeling of manipulated content and the visual continuity around the manipulated region.

For each dataset, 50% of the original videos were ramdomly selected, and applied the manipulation algorithm. These manipulated datasets were then prepared for model processing.

Fig. 3. The figure illustrates a sequence of frames extracted from the manipulated video *"deleted KATIE GETS HER DRIVERS PERMIT!! __first time driving_4_2"* from the DTD dataset. The frames are displayed chronologically, with their indices shown above each frame. A discontinuity in the temporal sequence is observed between the 228th and 229th frames, where 59 frames were omitted in the manipulated video. Consequently, the 229th frame has been marked as manipulated and is highlighted with a red border to emphasize this annotation.

IV. Experimental Settings

A. Hardware Used

The experiments were performed on a computer system with the following specifications: an Intel Core i9 12th generation processor, an MSI MAG B660M Mortar DDR4 motherboard, 64 GB of DDR4 RAM running at 3200 MHz, a 1TB NVMe solid-state drive, a 1TB SATA hard disk, and an MSI Nvidia GeForce RTX 3080 graphics card with 10GB of VRAM.

B. Datasets

The experiments were conducted on three primary datasets and their combinations, specifically: D1, D2, D3, D1+D2, D1+D3, D2+D3, and D1+D2+D3.

C. Metrics

It is inmportant to mention that four metrics to measure the behaviour of P1 and P2 were used, which are the following; accuracy (1), precision (3), recall (2) and F1 score (4).

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{2}$$

$$\text{Precision} = \frac{TP}{TP + FP} \tag{3}$$

$$F1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{4}$$

D. Training Parameters

Training of P1 was conducted for 100 epochs with the following parameters: a learning rate of 0.009, momentum of 0.9, and the use of the SGD optimizer.

TABLE I
EXPERIMENTAL SETTINGS FOR P2 SCENARIOS

Scenario	Seed	λ	δ	γ_1	γ_2
P2 [21]	40	0.1	2.6	1.4	1.0
P2 (pretrained networks)	–	0.08	3.5	1.4	1.0

In contrast, the settings for the **P2** experiments were divided into two scenarios, as shown in Table I. The first scenario represents the original work of [21], i.e., **P2** [21], while the second scenario includes five models based on [21], i.e., **P2** with pretrained networks. The seed refers to a random number used to initialize the weights of the CNN. For **P2** [21], the seed is set to 40, whereas in the second scenario, it is not required since it utilizes pretrained CNNs. For the other parameters (λ, δ, γ_1, γ_2), different values are required since each scenario applies distinct preprocessing and feature extraction techniques (see Section III).

V. RESULTS

Tables II and III present the performance of the seven models evaluated using 5-fold and 10-fold cross-validation, respectively. Each table reports the mean macro-average of the metrics across folds, along with their corresponding standard deviations. In general, a noticeable improvement in P1's performance can be observed in the 10-fold cross-validation compared to the 5-fold cross-validation. On the other hand, standard deviations tend to be lower in 5-fold cross-validation than in 10-fold cross-validation. This is due to the nature of the technique itself: with 5-fold cross-validation, each validation set contains more data, leading to a more stable and reliable model evaluation.

The key metric to focus on is recall, as it measures the model's ability to correctly identify true positives (videos with frame deletion). In Table II, the most reliable model for this task is P2 in D3, achieving a recall of 80%. Meanwhile, in Table III, P1 in D2+D3 stands out with a 93% recall, indicating the best performance in detecting frame deletion. Interestingly, P1 performed poorly on the D3 dataset, recording the lowest scores (33% on Recall) despite its strong performance in its combination with D2. Given that D3 is the smallest dataset, this limitation is likely the main cause of these results.

For most P2 models, results were consistent in Table III (10-fold cross-validation), except for P2-NasnetLarge and P2-InceptionV3, which performed poorly on D2. P2-NasnetLarge had the lowest scores: 25% Precision, 50% Recall, 33% F1-Score, and 50% Accuracy, while P2-InceptionV3 recorded 30%, 49%, 34%, and 50%, respectively. Notably, the original P2 model outperformed all variants, achieving the highest scores on D3: 81% Precision, 80% Recall, F1-Score, and Accuracy.

After applying hypothesis testing at a 5% significance level between the aforementioned models and supporting the analysis with the results shown in Table III, it is concluded that:

- With D1, P1 performs better.
- With D2, there is no significant difference between models.
- With D3, P2 has better performance.
- With D1+D2, P1 performs better.
- With D1+D3, P1 performs better.
- With D2+D3, P1 performs much better.
- With D1+D2+D3, P1 performs better.

In addition to classification performance, we report the model complexity and computational requirements for each approach. P1 contains approximately 9.97 million parameters, while P2 (original) contains 16 368, P2 - Resnet has 23.59 million parameters, P2 - VGG-16 contains 14.71, P2 - Nasnet has 84.92 million parameters, P2 - InceptionV3 contains 21.8 million parameters and P2 - DenseNet has 7 million of parameters. These differences in parameter count impact both training time and inference efficiency. About input resolution, the 3DCNN, from P1, was trained with an input resolution of (36, 64, 64, 3), while in P2, the original VGG model has (250, 250, 1) and the pretrained models (128, 128, 3). The average inference time per video sample was [insert time, e.g., 0.15s] for Model 1, [insert for Model 2], and [insert for Model 3]. These factors are considered further in the discussion section.

VI. DISCUSSION OF RESULTS

In Fig. 4, the ROC curves of the seven evaluated models on the combined dataset D1+D2+D3 (5-fold cross-validation) are presented. Based on the Area Under the Curve (AUC), P1 and P2 (VGG16) achieved the highest performance, both with an AUC of 0.64. P2 (Seed40) and P2 (ResNet50) followed closely, obtaining 0.62. On the other hand, P2 (NasnetLarge) recorded the lowest performance, with an AUC of 0.53, while P2 (DenseNet121) and P2 (InceptionV3) both reached 0.56. These results indicate that P1 and P2 (VGG16) are the most reliable models in distinguishing between positive and negative cases.

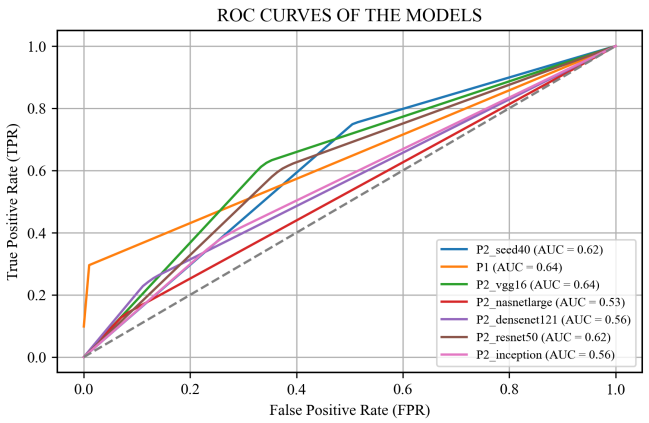


Fig. 4. A comparison of seven models using 5-fold cross-validation was conducted in terms of their ROC curves. Among all the models, P1 demonstrated the best performance when evaluated on D1+D2+D3, the largest dataset utilized in this study.

TABLE II

THE RESULTS OF EXPERIMENTS CONDUCTED WITH SEVEN MODELS ACROSS ALL DATASETS ARE PRESENTED. THE EXPERIMENTS WERE PERFORMED USING 5-FOLD CROSS-VALIDATION, AND EACH REPORTED VALUE REPRESENTS THE MEAN MACRO-AVERAGE ACROSS THE 5 FOLDS

Dataset	**Method**	**Metrics**							
		Precision	**std**	**Recall**	**std**	**F1**	**std**	**Accuracy**	**std**
D1	P1 [18](100 epochs)	0.76	±0.01	0.55	±0.03	0.44	±0.07	0.55	±0.03
	P2([21])	0.63	±0.00	0.62	±0.00	0.61	±0.00	0.62	±0.00
	P2(VGG-16 [38])	0.64	±0.01	0.64	±0.01	0.64	±0.01	0.64	±0.01
	P2(Resnet-50 [35])	0.62	±0.01	0.62	±0.01	0.62	±0.01	0.62	±0.01
	P2(NasnetLarge [37])	0.58	±0.02	0.53	±0.01	0.44	±0.00	0.53	±0.01
	P2(InceptionV3 [39])	0.57	±0.01	0.56	±0.01	0.55	±0.01	0.56	±0.01
	P2(Densenet-121 [36])	0.60	±0.01	0.56	±0.01	0.51	±0.01	0.56	±0.01
D2	P1 [18](100 epochs)	0.25	±0.00	0.50	±0.00	0.34	±0.00	0.50	±0.01
	P2([21])	0.57	±0.22	0.53	±0.08	0.45	±0.10	0.54	±0.08
	P2(VGG-16 [38])	0.72	±0.15	0.69	±0.12	0.68	±0.13	0.69	±0.12
	P2(Resnet-50 [35])	0.64	±0.08	0.64	±0.08	0.64	±0.08	0.64	±0.08
	P2(NasnetLarge [37])	0.25	±0.00	0.50	±0.00	0.34	±0.00	0.50	±0.01
	P2(InceptionV3 [39])	0.35	±0.21	0.49	±0.03	0.35	±0.04	0.50	±0.04
	P2(Densenet-121 [36])	0.57	±0.26	0.55	±0.07	0.43	±0.11	0.55	±0.07
D3	P1 [18](100 epochs)	0.25	±0.01	0.50	±0.0	0.33	±0.00	0.50	±0.01
	P2([21])	**0.81**	**±0.06**	**0.80**	**±0.07**	**0.80**	**±0.07**	**0.80**	**±0.06**
	P2(VGG-16 [38])	0.76	±0.06	0.74	±0.07	0.73	±0.08	0.74	±0.07
	P2(Resnet-50 [35])	0.71	±0.04	0.70	±0.05	0.69	±0.05	0.70	±0.05
	P2(NasnetLarge [37])	0.60	±0.10	0.53	±0.03	0.44	±0.04	0.53	±0.03
	P2(InceptionV3 [39])	0.62	±0.08	0.61	±0.08	0.60	±0.08	0.61	±0.08
	P2(Densenet-121 [36])	0.61	±0.03	0.58	±0.03	0.56	±0.04	0.58	±0.03
D1+D2	P1 [18](100 epochs)	0.76	±0.01	0.53	±0.02	0.40	±0.03	0.53	±0.02
	P2([21])	0.63	±0.01	0.62	±0.01	0.61	±0.01	0.62	±0.01
	P2(VGG-16 [38])	0.64	±0.01	0.64	±0.01	0.64	±0.01	0.64	±0.01
	P2(Resnet-50 [35])	0.62	±0.01	0.62	±0.01	0.62	±0.01	0.62	±0.01
	P2(NasnetLarge [37])	0.58	±0.02	0.53	±0.01	0.44	±0.01	0.53	±0.01
	P2(InceptionV3 [39])	0.57	±0.01	0.56	±0.01	0.55	±0.01	0.56	±0.01
	P2(Densenet-121 [36])	0.60	±0.02	0.56	±0.01	0.51	±0.01	0.56	±0.01
D1+D3	P1 [18](100 epochs)	0.76	±0.02	0.56	±0.03	0.45	±0.05	0.56	±0.03
	P2([21])	0.63	±0.00	0.62	±0.00	0.62	±0.01	0.62	±0.00
	P2(VGG-16 [38])	0.64	±0.01	0.64	±0.01	0.64	±0.01	0.64	±0.01
	P2(Resnet-50 [35])	0.62	±0.01	0.62	±0.01	0.62	±0.01	0.62	±0.01
	P2(NasnetLarge [37])	0.58	±0.01	0.53	±0.00	0.44	±0.01	0.53	±0.00
	P2(InceptionV3 [39])	0.57	±0.01	0.56	±0.01	0.55	±0.02	0.56	±0.01
	P2(Densenet-121 [36])	0.60	±0.01	0.56	±0.01	0.51	±0.01	0.56	±0.01
D2+D3	P1 [18](100 epochs)	0.25	±.00	0.50	±0.00	0.33	±0.00	0.50	±0.00
	P2([21])	0.70	±.05	0.67	±0.06	0.66	±0.06	0.68	±0.05
	P2(VGG-16 [38])	0.72	±0.07	0.72	±0.07	0.71	±0.08	0.72	±0.08
	P2(Resnet-50 [35])	0.68	±0.10	0.67	±0.10	0.67	±0.10	0.67	±0.10
	P2(NasnetLarge [37])	0.54	±0.17	0.51	±0.02	0.40	±0.04	0.52	±0.02
	P2(InceptionV3 [39])	0.57	±0.10	0.56	±0.06	0.51	±0.09	0.56	±0.06
	P2(Densenet-121 [36])	0.49	±0.15	0.52	±0.05	0.43	±0.08	0.51	±0.05
D1+D2+D3	P1 [18](100 epochs)	0.79	±0.02	0.64	±0.06	0.59	±0.08	0.64	±0.06
	P2([21])	0.63	±0.01	0.62	±0.00	0.62	±0.00	0.62	±0.00
	P2(VGG-16 [38])	0.64	±0.01	0.64	±0.01	0.64	±0.01	0.64	±0.01
	P2(RESNET-50 [35])	0.62	±0.01	0.62	±0.01	0.62	±0.01	0.62	±0.01
	P2(NasnetLarge [37])	0.58	±0.01	0.53	±0.00	0.44	±0.00	0.53	±0.00
	P2(InceptionV3 [39])	0.57	±0.01	0.56	±0.01	0.55	±0.01	0.56	±0.01
	P2(Densenet-121 [36])	0.60	±0.02	0.56	±0.01	0.51	±0.01	0.56	±0.01

On the other hand, for 10-fold cross-validation experiments, in Fig. 5, P1 performs the best compared with the other six models. P1 obtained 0.77 in terms of AUC, P2 (VGG16) obtained 0.64, being the highest within P2 variants. Then, P2 (Original) achieved 0.62, P2 (NasnetLarge) obtained 0.53, being the lowest among P2 models, P2 (DenseNet121) obtained 0.56, P2 (ResNet50) obtained 0.62, and finally P2 (InceptionV3) obtained 0.56.

As it is shown in Table IV, P2 (Original) obtained the best average inference time (0.73 s/min) despite its larger input resolution compared to other P2 models. On the other hand, P2 (Nasnet) exhibits the highest computational cost (9.73 s/min). Notably, Model P1, which processes the largest input resolution, ranks second in efficiency (1.61 s/min). Variability in performance highlights differences across models, underlining the impact of architecture design on efficiency.

Why do some results from P2 show poor performance, with accuracy and other metrics falling within the range of 50-60%? This decline can be attributed to the frames and their statistical analysis shown in Figs. 6 and 7. First, in case of Fig. 6, when a

TABLE III

THE RESULTS OF EXPERIMENTS CONDUCTED WITH SEVEN MODELS ACROSS ALL DATASETS ARE PRESENTED. THE EXPERIMENTS WERE PERFORMED USING 10-FOLD CROSS-VALIDATION, AND EACH REPORTED VALUE REPRESENTS THE MEAN MACRO-AVERAGE ACROSS THE 10 FOLDS.

Dataset	**Method**	**Metrics**							
		Precision	**std**	**Recall**	**std**	**F1**	**std**	**Accuracy**	**std**
D1	P1 [18](100 epochs)	0.89	±0.09	0.82	±0.18	0.78	±0.23	0.82	±0.18
	P2([21])	0.63	±0.01	0.62	±0.01	0.61	±0.01	0.62	±0.01
	P2(VGG-16 [38])	0.64	±0.01	0.64	±0.01	0.64	±0.01	0.64	±0.01
	P2(Resnet-50 [35])	0.62	±0.01	0.62	±0.01	0.62	±0.01	0.62	±0.01
	P2(NasnetLarge [37])	0.58	±0.02	0.53	±0.01	0.44	±0.01	0.53	±0.01
	P2(InceptionV3 [39])	0.57	±0.01	0.56	±0.01	0.55	±0.01	0.56	±0.01
	P2(Densenet-121 [36])	0.60	±0.02	0.56	±0.01	0.51	±0.01	0.56	±0.01
D2	P1 [18](100 epochs)	0.88	±0.23	0.88	±0.18	0.86	±0.22	0.88	±0.20
	P2([21])	0.51	±0.29	0.53	±0.13	0.44	±0.16	0.54	±0.14
	P2(VGG-16 [38])	0.71	±0.21	0.69	±0.15	0.67	±0.17	0.69	±0.14
	P2(Resnet-50 [35])	0.65	±0.13	0.64	±0.12	0.64	±0.12	0.64	±0.12
	P2(NasnetLarge [37])	0.25	±0.02	0.50	±0.00	0.33	±0.02	0.50	±0.04
	P2(InceptionV3 [39])	0.30	±0.16	0.49	±0.04	0.34	±0.06	0.50	±0.06
	P2(Densenet-121 [36])	0.52	±0.27	0.55	±0.07	0.43	±0.11	0.55	±0.07
D3	P1 [18](100 epochs)	0.25	±0.01	0.50	±0.0	0.33	±0.01	0.50	±0.01
	P2([21])	**0.81**	**±0.08**	**0.80**	**±0.08**	**0.80**	**±0.08**	**0.80**	**±0.08**
	P2(VGG-16 [38])	0.76	±0.12	0.74	±0.11	0.73	±0.11	0.74	±0.11
	P2(Resnet-50 [35])	0.72	±0.05	0.70	±0.06	0.69	±0.07	0.70	±0.06
	P2(NasnetLarge [37])	0.55	±0.22	0.53	±0.08	0.44	±0.10	0.53	±0.07
	P2(InceptionV3 [39])	0.63	±0.12	0.61	±0.10	0.60	±0.10	0.61	±0.10
	P2(Densenet-121 [36])	0.61	±0.11	0.58	±0.09	0.55	±0.10	0.58	±0.08
D1+D2	P1 [18](100 epochs)	0.85	±0.07	0.76	±0.14	0.73	±0.19	0.76	±0.14
	P2([21])	0.63	±0.01	0.62	±0.01	0.61	±0.01	0.62	±0.01
	P2(VGG-16 [38])	0.64	±0.01	0.64	±0.01	0.64	±0.01	0.64	±0.01
	P2(Resnet-50 [35])	0.62	±0.02	0.62	±0.02	0.62	±0.02	0.62	±0.02
	P2(NasnetLarge [37])	0.58	±0.03	0.53	±0.01	0.44	±0.01	0.53	±0.01
	P2(InceptionV3 [39])	0.57	±0.01	0.56	±0.01	0.55	±0.01	0.56	±0.01
	P2(Densenet-121 [36])	0.60	±0.02	0.56	±0.01	0.51	±0.01	0.56	±0.01
D1+D3	P1 [18](100 epochs)	0.86	±0.07	0.78	±0.15	0.75	±0.19	0.78	±0.15
	P2([21])	0.63	±0.01	0.62	±0.01	0.62	±0.01	0.62	±0.01
	P2(VGG-16 [38])	0.64	±0.01	0.64	±0.01	0.64	±0.01	0.64	±0.01
	P2(Resnet-50 [35])	0.62	±0.01	0.62	±0.01	0.62	±0.01	0.62	±0.01
	P2(NasnetLarge [37])	0.58	±0.02	0.53	±0.01	0.44	±0.01	0.53	±0.01
	P2(InceptionV3 [39])	0.57	±0.02	0.56	±0.01	0.55	±0.02	0.56	±0.01
	P2(Densenet-121 [36])	0.60	±0.02	0.56	±0.01	0.51	±0.01	0.56	±0.01
D2+D3	P1 [18](100 epochs)	**0.94**	**±0.12**	**0.93**	**±0.14**	**0.92**	**±0.15**	**0.93**	**±0.14**
	P2([21])	0.70	±.01	0.67	±0.01	0.66	±0.01	0.67	±0.01
	P2(VGG-16 [38])	0.72	±0.08	0.72	±0.08	0.71	±0.09	0.72	±0.08
	P2(Resnet-50 [35])	0.68	±0.10	0.67	±0.10	0.67	±0.10	0.67	±0.10
	P2(NasnetLarge [37])	0.48	±0.21	0.52	±0.04	0.40	±0.07	0.52	±0.05
	P2(InceptionV3 [39])	0.56	±0.13	0.56	±0.06	0.51	±0.10	0.56	±0.07
	P2(Densenet-121 [36])	0.60	±0.16	0.57	±0.07	0.50	±0.11	0.57	±0.08
D1+D2+D3	P1 [18](100 epochs)	0.85	±0.07	0.77	±0.13	0.74	±0.17	0.77	±0.13
	P2([21])	0.63	±0.01	0.62	±0.01	0.62	±0.01	0.62	±0.01
	P2(VGG-16 [38])	0.64	±0.01	0.64	±0.01	0.64	±0.01	0.64	±0.01
	P2(RESNET-50 [35])	0.62	±0.02	0.62	±0.02	0.62	±0.02	0.62	±0.02
	P2(NasnetLarge [37])	0.58	±0.02	0.53	±0.01	0.44	±0.01	0.53	±0.01
	P2(InceptionV3 [39])	0.57	±0.02	0.56	±0.02	0.55	±0.01	0.56	±0.02
	P2(Densenet-121 [36])	0.60	±0.02	0.56	±0.01	0.51	±0.01	0.56	±0.01

forged video is predicted as original, it could happen because the deletion point was in a static or low motion point, i.e. almost no object movement was detected by P2 and thus is considered wrongly original.

In case of Fig. 7, when a original video is predicted as forged, two abrupt changes are detected by PCC than is consider as deletion points, thus is considered wrongly forged.

VII. CONCLUSIONS

In this work, various scenarios to evaluate both a supervised (P1) and unsupervised (P2) approach have been explored, evaluating a total of seven models (a P1 model and six P2 variants) for frame deletion detection. Apart from using two datasets from literature (UCF101 and VIFFD), the DTD dataset is presented. These datasets were finally combined, composing a total of seven datasets and apply 5-fold and 10-fold cross-validation for the testing stage. This work highlights that P1 model surpassed the P2 variants in almost all datasets,

TABLE IV
COMPUTATIONAL COST OF EACH MODEL

Model	# Parameters	Input Resolution	Average Inference Time (sec/min)
P1 - 3DCNN	9.97M	$36 \times 64 \times 64 \times 3$	1.61
P2 - Original	16,368M	$250 \times 250 \times 1$	0.73
P2 - Resnet50	23.59M	$128 \times 128 \times 3$	2.65
P2 - VGG-16	14.71M	$128 \times 128 \times 3$	2.19
P2 - Nasnet	84.92M	$128 \times 128 \times 3$	9.73
P2 - InceptionV3	21.8M	$128 \times 128 \times 3$	5.04
P2 - Densenet	7M	$128 \times 128 \times 3$	8.63

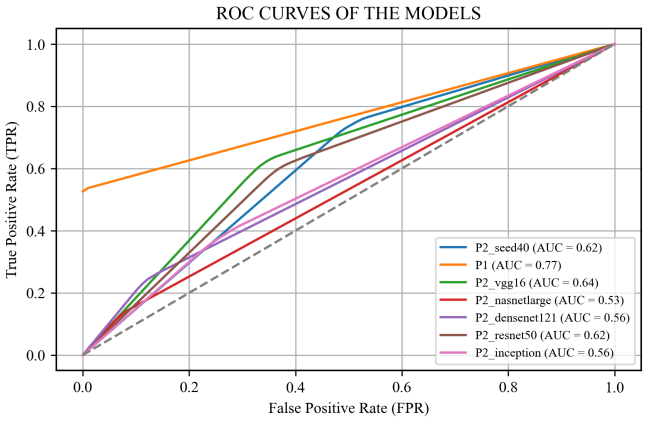


Fig. 5. A comparison of seven models using 10-fold cross-validation was conducted in terms of their ROC curves. Among all the models, P1 demonstrated the best performance when evaluated on D1+D2+D3, the largest dataset utilized in this study.

except for D3, in which P2 variants achieved the best values.

CNN is considered to be one of the most relevant techniques today, showing significant results in this study. However, it is still necessary to address this task with techniques that can potentially increase performance, particularly in terms of the F1 score. Seven models in total have been succesfully compared under the same scenarios to gain a deeper understanding of their performance across datasets from both the literature and the developed dataset. Although the P1 model consistently outperformed the six P2 variants in terms of AUC score—demonstrating its reliability with a score of 0.77 in 10-fold cross-validation—it achieved the same highest score as P2 (VGG16) in the 5-fold cross-validation, both reaching 0.64. Regarding computational cost, P1 was the second most efficient model, with an inference time of 1.61 sec/min, just behind the 0.73 sec/min of the original P2, despite processing higher-resolution inputs. On the other hand, among the P2 variants, VGG16 proved to be the most competitive, achieving high AUC scores of 0.64 in both 5-fold and 10-fold cross-validation. However, other variants such as NasnetLarge, Densenet121, Resnet50 and Inception obtained suboptimal results not only in AUC (50-60%), but also in computational cost metrics. This can be suggests misclassifications in low-motion scenes or abrupt content changes.

For future research, it would be interesting to enhance forgery detection by exploring novel models such as Transformers. Although they have not yet been applied specifically to frame deletion detection, they hold great potential, particularly with variants like Vision Transformers (ViTs), which have demonstrated strong performance in various computer vision tasks.

ACKNOWLEDGMENTS

This work was financially supported by *Vicerrectorado de Investigación* (VRI) of Universidad San Ignacio de Loyola in the form of a grant awarded to this group.

REFERENCES

[1] S. Fadl, Q. Han, and Q. Li, "CNN spatiotemporal features and fusion for surveillance video forgery detection," *Signal Processing: Image Communication*, vol. 90, p. 116066, Jan. 2021, doi: 10.1016/j.image.2020.116066.

[2] M. Munawar, I. Noreen, R. S. Alharthi, and N. Sarwar, "Forged Video Detection Using Deep Learning: A SLR," *Applied Computational Intelligence and Soft Computing*, vol. 2023, p. e6661192, Oct. 2023, doi: 10.1155/2023/6661192.

[3] A. Yadav and D. Kumar Vishwakarma, "Datasets, clues and state-of-the-arts for multimedia forensics: An extensive review," *Expert Systems with Applications*, vol. 249, p. 123756, Sep. 2024, doi: 10.1016/j.eswa.2024.123756.

[4] P. Johnston and E. Elyan, "A review of digital video tampering: From simple editing to full synthesis," *Digital Investigation*, vol. 29, pp. 67–81, Jun. 2019, doi: 10.1016/j.diin.2019.03.006.

[5] G. Mercier, F. Markatopoulou, R. Cozien, M. Zampoglou, E. Apostolidis, A. Metsai, S. Papadopoulos, V. Mezaris, I. Patras, and I. Kompatsiaris, "Detecting manipulations in video," in *Video Verif. in the Fake News Era*. Springer International Publishing, 2019, pp. 161–189, journal Abbreviation: Video Verif. in the Fake News Era, doi: 10.1007/978-3-030-26752-0_6.

[6] L. Jegaveerapandian, A. Rani, P. Periyaswamy, and S. Velusamy, "A survey on passive digital video forgery detection techniques," vol. 13, no. 6, pp. 6324–6334, 2023, publisher: Institute of Advanced Engineering and Science, doi: 10.11591/ijece.v13i6.pp6324-6334.

[7] A. R. Javed, Z. Jalil, W. Zehra, T. Gadekallu, D. Suh, and M. Jalil Piran, "A comprehensive survey on digital video forensics: Taxonomy, challenges, and future directions," *Engineering Applications of Artificial Intelligence*, vol. 106, p. 104456, Nov. 2021, doi: 0.1016/j.engappai.2021.104456.

[8] N. A. Shelke and S. S. Kasana, "A comprehensive survey on passive techniques for digital video forgery detection," *Multimedia Tools and Applications*, vol. 80, no. 4, pp. 6247–6310, Feb. 2021, doi: 10.1007/s11042-020-09974-4.

[9] W. El-Shafai, M. A. Fouda, E.-S. M. El-Rabaie, and N. A. El-Salam, "A comprehensive taxonomy on multimedia video forgery detection techniques: challenges and novel trends," *Multimedia Tools and Applications*, May 2023, doi: 10.1007/s11042-023-15609-1.

[10] R. Gowda and D. Pawar, "Deep learning-based forgery identification and localization in videos," *Signal, Image and Video Processing*, vol. 17, no. 5, pp. 2185–2192, Jul. 2023, doi: 10.1007/s11760-022-02433-7.

[11] N. Akhtar, M. Saddique, K. Asghar, U. Bajwa, M. Hussain, and Z. Habib, "Digital Video Tampering Detection and Localization: Review, Representations, Challenges and Algorithm," *Mathematics*, vol. 10, no. 2, 2022, doi: 10.3390/math10020168.

[12] S. Ferreira, M. Antunes, and M. Correia, "A dataset of photos and videos for digital forensics analysis using machine learning processing," *Data*, vol. 6, no. 8, 2021, doi: 10.3390/data6080087.

[13] X. Nguyen, Y. Hu, M. Amin, K. Hayat, V. Le, and D.-T. Truong, "Detecting Video Inter-Frame Forgeries Based on Convolutional Neural Network Model," *International Journal of Image, Graphics and Signal Processing*, vol. 12, no. 3, pp. 1–12, 2020, doi: 10.5815/ijigsp.2020.03.01.

[14] H. Yi Gong, F. Chun Hui, and B. Dan Dan, "IReF: Improved Residual Feature For Video Frame Deletion Forensics," in *2022 4th International Conference on Data Intelligence and Security (ICDIS)*, Aug. 2022, pp. 248–253, doi: 10.1109/ICDIS55630.2022.00045.

[15] J. H. Hong, Y. Yang, and B. T. Oh, "Detection of frame deletion in HEVC-Coded video in the compressed domain," *Digital Investigation*, vol. 30, pp. 23–31, Sep. 2019, doi: 10.1016/j.diin.2019.06.002.

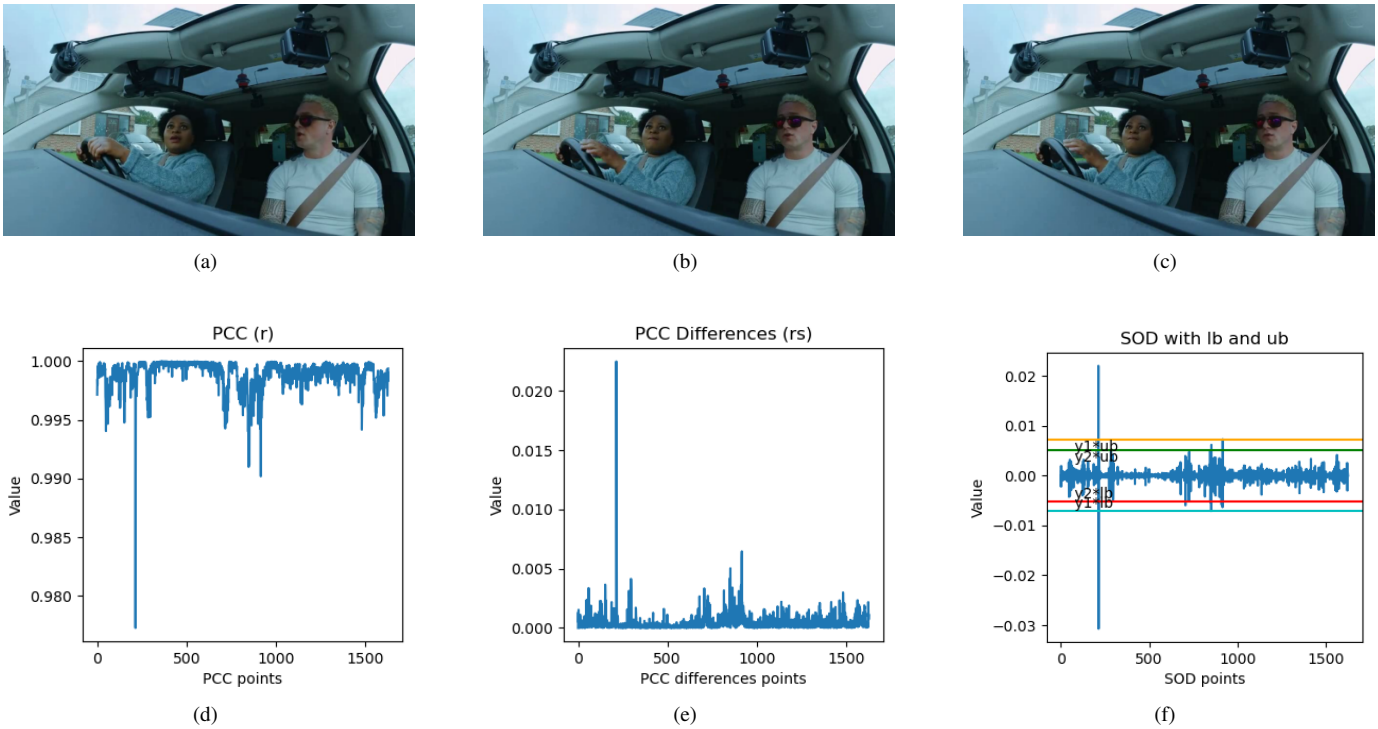


Fig. 6. **False Negative Scenario** Top row (a – c) displays consecutive forged frames; bottom row (d – f) shows the corresponding analysis plots. This forged video is part of the D3 dataset. Despite containing a forgery at the 7th second (between a and b), the P2 (VGG16) model classified the video as original. Actually, this forgery is reflected in the abrupt change around the 250th PCC Difference point (e), but it did not exceed the lambda threshold (λ = 0.08), and thus was not flagged as a deletion point (see I).

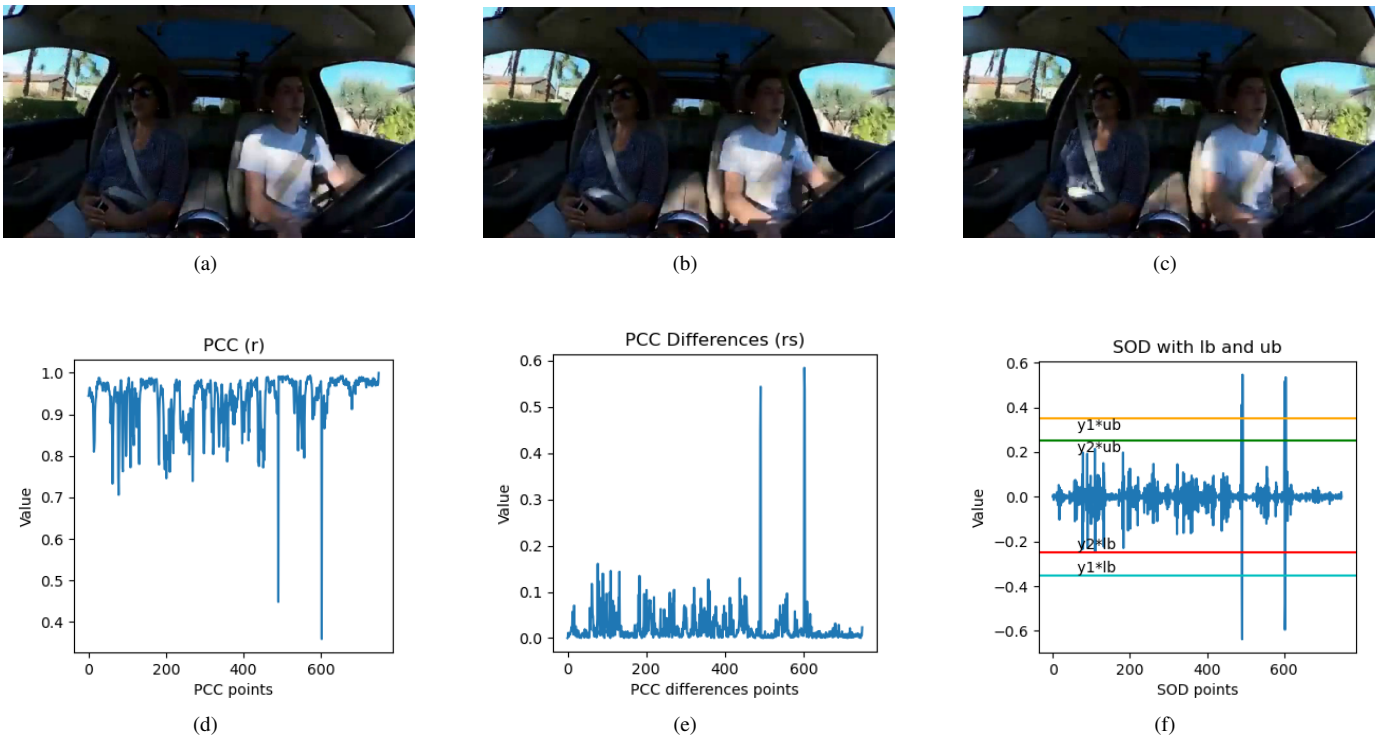


Fig. 7. **False Positive Scenario** Top row (a – c) displays consecutive orignal frames; bottom row ((d – f)) shows the corresponding analysis plots. This original video is from the D3 dataset. It was predicted as forged just between a and b, despite there is any cut. The abrupt changes in the lighting between frames reflect also in all three PCC plots (d - f) and, specifically, in the 600th point, explaining why the model mistakenly identified the video as forged.

[16] C. Long, E. Smith, A. Basharat, and A. Hoogs, "A C3D-Based Convolutional Neural Network for Frame Dropping Detection in a Single

Video Shot," in *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. Honolulu, HI, USA: IEEE, Jul. 2017, pp. 1898–1906, doi: 10.1109/CVPRW.2017.237.

[17] N. Singla, J. Singh, and S. Nagpal, "Video Frame Deletion Detection using Correlation Coefficients," in *2021 8th International Conference on Signal Processing and Integrated Networks (SPIN)*. Noida, India: IEEE, Aug. 2021, pp. 796–799, doi: 10.1109/SPIN52536.2021.9565979.

[18] M. Oraibi and A. Radhi, "Enhancement Digital Forensic Approach for Inter-Frame Video Forgery Detection Using a Deep Learning Technique," *Iraqi Journal of Science*, vol. 63, no. 6, pp. 2686–2701, 2022, doi: 10.24996/ijs.2022.63.6.34.

[19] J. Deng, W. Dong, R. Socher, L.-J. Li, Kai Li, and Li Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2009, pp. 248–255, doi: 10.1109/CVPR.2009.5206848.

[20] M. Saddique, K. Asghar, U. Bajwa, M. Hussain, H. Aboalsamh, and Z. Habib, "Classification of authentic and tampered video using motion residual and parasitic layers," *IEEE Access*, vol. 8, pp. 56 782–56 797, 2020, doi: 10.1109/ACCESS.2020.2980951.

[21] V. Kumar, V. Kansal, and M. Gaur, "Multiple Forgery Detection in Video Using Convolution Neural Network," *Computers, Materials and Continua*, vol. 73, no. 1, pp. 1347–1364, 2022, doi: 10.32604/cmc.2022.023545.

[22] A. Bavrina, "Method for Frame Removal Detection in Static Camera Surveillance Video," in *Proc. - IEEE Int. Conf. Inf. Technol. Nanotechnol., ITNT*. Institute of Electrical and Electronics Engineers Inc., 2023, doi: 10.1109/ITNT57377.2023.10139053.

[23] Q. Abbas, M. E. A. Ibrahim, and M. A. Jaffar, "Video scene analysis: an overview and challenges on deep learning algorithms," *Multimedia Tools and Applications*, vol. 77, no. 16, pp. 20 415–20 453, Aug. 2018, doi: 10.1007/s11042-017-5438-7.

[24] S. Ferreira, M. Antunes, and M. E. Correia, "Exposing Manipulated Photos and Videos in Digital Forensics Analysis," *Journal of Imaging*, vol. 7, no. 7, p. 102, Jul. 2021, doi: 10.3390/jimaging7070102.

[25] X. Jin, Y. Su, and P. Jing, "Video frame deletion detection based on time–frequency analysis," *Journal of Visual Communication and Image Representation*, vol. 83, p. 103436, Feb. 2022, doi: 10.1016/j.jvcir.2022.103436.

[26] X. H. Nguyen, "VIFFD - a dataset for detecting video inter-frame forgeries," 2020, doi: 10.17632/R3SS3V53SJ.6.

[27] A. R. Z. Khurram Soomro and M. Shah, "Ucf101: A dataset of 101 human action classes from videos in the wild." in *CRCV*, 2012, doi: 10.48550/arXiv.1212.0402.

[28] C. Tinipuclla, J. Ceron, and P. Shiguihara, "Frame Deletion Detection in Videos Using Convolutional Neural Networks," in *2024 IEEE ANDESCON*. Cusco, Peru: IEEE, Sep. 2024, pp. 1–6, doi: 10.1109/ANDESCON61840.2024.10755836.

[29] L. Yu, H. Wang, Q. Han, X. Niu, S. Yiu, J. Fang, and Z. Wang, "Exposing frame deletion by detecting abrupt changes in video streams," *Neurocomputing*, vol. 205, pp. 84–91, Sep. 2016, doi: 10.1016/j.neucom.2016.03.051.

[30] C. Feng, Z. Xu, S. Jia, W. Zhang, and Y. Xu, "Motion-Adaptive Frame Deletion Detection for Digital Video Forensics," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 27, no. 12, pp. 2543–2554, Dec. 2017, doi: 10.1109/TCSVT.2016.2593612.

[31] C. C. Huang, C. E. Lee, and V. L. L. Thing, "A Novel Video Forgery Detection Model Based on Triangular Polarity Feature Classification:," *International Journal of Digital Crime and Forensics*, vol. 12, no. 1, pp. 14–34, Jan. 2020, doi: 10.4018/IJDCF.2020010102.

[32] Y. Wang, Y. Hu, A. W.-C. Liew, and C.-T. Li, "ENF Based Video Forgery Detection Algorithm:," *International Journal of Digital Crime and Forensics*, vol. 12, no. 1, pp. 131–156, Jan. 2020, doi: 10.4018/IJDCF.2020010107.

[33] S. Li and H. Huo, "Frame Deletion Detection Based on Optical Flow Orientation Variation," *IEEE Access*, vol. 9, pp. 37 196–37 209, 2021, doi: 10.1109/ACCESS.2021.3061586.

[34] J. Bakas, R. Naskar, and S. Bakshi, "Detection and localization of inter-frame forgeries in videos based on macroblock variation and motion vector analysis," *Computers & Electrical Engineering*, vol. 89, p. 106929, Jan. 2021, doi: 10.1016/j.compeleceng.2020.106929.

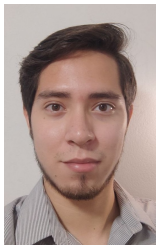
[35] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015, doi: 10.48550/arXiv.1512.03385.

[36] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," 2018, doi: 10.48550/arXiv.1608.06993.

[37] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le, "Learning transferable architectures for scalable image recognition," 2018, doi: 10.48550/arXiv.1707.07012.

[38] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2015, doi: 10.48550/arXiv.1409.1556.

[39] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," 2015, doi: 10.48550/arXiv.1512.00567.



Jorge Ceron received his Bachelor's degree in Computer Science and Systems Engineering from Universidad San Ignacio de Loyola (USIL), Lima, Peru. He is currently a research member of the Computer Science Research Group (GICC) at USIL. His research interests include machine learning and computer vision.

Cristian Tinipuclla received the B.S. degree in Systems and Informatics Engineering from Universidad San Ignacio de Loyola (USIL), Peru, in 2022. He is currently a research member of the Computer Science Research Group (GICC) at USIL. His research interests include computer vision and neural networks.

Pedro Shiguihara (Senior Member, IEEE) received a master's degree from the University of São Paulo, in 2013. He is currently pursuing a Ph.D. degree with the National University of San Marcos. He heads the Computer Science Research Group of the Universidad San Ignacio de Loyola, Lima, Peru.