Journal of Information Systems Engineering and Management
2025, 10(29s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Exposing Video Forgeries: A Deep Learning Analysis of SVM vs. CNN

Hemant Appa Tirmare[1],[2]*, Jaydeep B. Patil[2], Vidyullata Vinayak Devmane[3], Shashikant Sudhakar Radke[3], Rita Khillare kadam[4]

[1]*Department of Technology, Shivaji University, Kolhapur, Maharashtra, India- 416004*
[2]*Department of Computer Science Engineering, D. Y. Patil Agriculture and Technical University, Talsande, Maharashtra, India-416112*
[3]*Department of Computer Engineering, Shah and Anchor Kutchhi Engineering College, Mumbai, Maharashtra, India – 400088.*
[4]*Department of Artificial Intelligence and Data science, D Y Patil University Pune, Ambi, Talegaon Dabhade, Maharashtra,India- 410506*
**Corresponding author email: hat_tech@unishivaji.ac.in*

ARTICLE INFO

Received: 29 Dec 2024

Revised: 12 Feb 2025

Accepted: 27 Feb 2025

ABSTRACT

Introduction: The rapid proliferation of digital media has heightened concerns regarding video forgeries, necessitating robust detection mechanisms for security and forensic applications. Traditional methods often struggle to detect complex forgeries, particularly in low-quality or compressed videos.

Objectives: This study conducts an analysis of Support Vector Machines (SVM) and Convolutional Neural Networks (CNN) for video forgery detection. The dataset underwent preprocessing, including resizing, grayscale conversion, and normalization, to enhance frame uniformity and reduce noise. Videos were segmented, and the difference of consecutive frames (DOCFs) was computed to extract feature vectors for SVM classification..

Methods: CNN was trained to detect complex spatial-temporal forgery patterns, including edge inconsistencies and visual artifacts, by utilizing convolutional and pooling layers, which were analyzed by fully connected layers.

Results: Experimental results demonstrated that SVM achieved an accuracy of 46.67%, while CNN significantly outperformed it with an accuracy of 73.33% on a dataset of 57 videos. Further analysis on a smaller dataset of 33 videos showed CNN achieving 81.82% accuracy, reinforcing its adaptability and robustness in detecting forgeries..

Conclusions: The study highlights CNN's superiority in handling intricate manipulations, positioning it as a viable approach for scalable and precise video forgery detection in evolving digital landscapes.

Keywords: Video forgery detection, Digital media forgeries, Support Vector Machine, Convolutional Neural Networks, Difference of Consecutive frames, Machine learning, Deep learning, forgery localization, feature extraction..

INTRODUCTION

The widespread distribution of video content on several platforms in modern times has sparked serious questions about the authenticity and reliability of these visual representations. Video forgery, manipulation or alteration of video footage, has emerged as a critical challenge with far-reaching implications for individual privacy, societal trust, and the dissemination of accurate information [1]. The increasing availability of advanced digital editing tool has facilitated the creation of convincing video forgeries, commonly referred to as “deepfakes” [2]. These manipulated videos, images, audio, or text are often employed to spread misinformation, manipulate public opinion, and perpetrate malicious activities such as financial fraud or revenge porn [3]. The growing sophistication of these technologies necessitates the development of effective detection methods. Digital videos serve as valuable evidence and hold significant importance across various fields, including media organizations, insurance companies, marriage bureaus, investigation agencies, social media platforms [4], and legal systems. However, with the accessibility of video editing software includes Adobe Premiere, After Effects, GNU Gimp, and Vegas, tampering with videos has

become easier, leading to their widespread misuse on social media and in legal contexts. This undermines the credibility of videos and highlights the need for robust forensic analysis to verify their authenticity and integrity [5].

Techniques for detecting video forgeries can be broadly divided into two categories: active and passive [6]. Active methods rely on embedded information includes watermarks or digital signs, while passive approaches authenticate videos without requiring any prior information, making them more versatile and widely applicable [7]. Passive forgery detection, particularly in the spatial domain, has become a prominent research area in information security. Spatial domain video forgery occurs through methods like copy-move forgery, where an object is duplicated within the same video, and splicing, where an object from one video is introduced into another [8]. Despite advancements in image forgery detection, identifying spatial or object-based forgery in videos remains challenging due to video compression, high computational complexity, and the appearance of forgery traces across consecutive frames.

Spatial domain tampering often alters the texture of micro-patterns in video frames, providing strong indicators for detection. Existing descriptors [9] includes Histogram of Oriented Gradients (HOG) and Local Binary Patterns (LBP) may be used for forgery detection. While HOG captures gradient orientation, it focuses on shape information and is vulnerable to noise and scale variations. Similarly, LBP is robust against monotonic illumination changes but is sensitive to noise and small gray-level fluctuations. These limitations necessitate the development of a more reliable descriptor for detecting spatial forgery in videos. The increasing sophistication and accessibility of video editing tools, coupled with the rise of deepfake technologies, have amplified the challenge of maintaining the authenticity and integrity of video content [10]. This widespread manipulation poses significant threats to privacy, societal trust, and the credibility of digital media, necessitating robust and effective detection techniques [11]. By focusing on spatial domain tampering and leveraging innovative feature descriptors and classification methods, this research purposes to bridge the existing gaps in video forgery detection, paving the way for more reliable and efficient authentication frameworks.

Contributions of the research

This study proposes an approach to video forgery detection by using both ML and DL techniques. The main contributions of this study are as:

- The study pre-processes video datasets by segmenting videos into frames, applying resizing, grayscale conversion, and normalization to reduce noise and ensure uniformity across frames.
- Videos are divided into segments, and DOCFs are computed to extract feature vectors indicative of tampering.
- An SVM classifier is trained on DOCFs to classify video segments. If at least one pair of consecutive frames within a segment shows signs of forgery, the entire segment is flagged as forged; otherwise, it is deemed authentic.
- CNN model is developed to identify intricate forgery features, such as inconsistencies in object edges, unnatural textures, or visual artifacts introduced during splicing or copy-move manipulations. The CNN not only classifies frames as “authentic” or “forged” but also localizes tampered regions within frames.
- The methods are evaluated on datasets of forged and authentic videos, demonstrating that CNN outperforms SVM in terms of accuracy, precision, particularly in handling intricate forgery scenarios.

Organization of the work

This work is organized as: section 1 offers a summary of video forgery detection, and the need for effective detection methods. Section 2 reviews existing studies and their methodologies, in addressing video forgery. Section 3 outlines the proposed methodology for classifying videos as authentic or forged and for identifying tampered frames. Section 4 explains the outcomes of the proposed method, including its performance evaluation and comparative analysis. The work is concluded in Section 5, which also offers possible areas for further research while summarizing the main findings.

LITERATURE SURVEY

Farukh Hashmi et al. [12] proposed a Conv-LSTM-based benchmark architecture that leverages facial landmarks and convolutional features to automatically identify visual forgery in images and videos. The authors highlighted that their approach achieves precise accuracy in detecting visual forgeries, surpassing existing state-of-the-art methods.

This architecture was trained on an extensive dataset, enabling it to effectively identify visual forgery with high accuracy. Despite the inherent complexity of the black-box training process within neural networks, the suggested method addresses memory constraints by eliminating the need to save video frames during training. Although this approach is expected to be significantly more complex, it opens pathways for the development of lightweight models tailored for forgery detection.

Minh Dang et al. [13] introduced a DL-based framework designed to accurately detect manipulated face images. A specific convolutional neural network (CNN) system for Manipulated Face (MANFA) detection was suggested by the study. Multiple convolutional layers are used in this framework to efficiently extract characteristics from tampered regions at various abstraction levels. A hybrid framework (HF-MANFA) that combines Adaptive Boosting (AdaBoost) and eXtreme Gradient Boosting (XGBoost) was presented to address issues related to imbalanced datasets. To increase framework's dependability, a thorough dataset of modified faces was also manually gathered and verified. According to experimental results, the suggested models performed better than current expert and AI-driven systems in terms of computational efficiency, resilience to data imbalance, and area under the curve (AUC). Minh Dang et al. concluded that the presented framework provides a significant step forward in the detection of altered face images. They emphasized its potential for motivating the development of more advanced detection methods and its applicability in systems regularly dealing with manipulated images.

Sabir et al [14] discuss the growing challenge of detecting misinformation through synthetically generated images and videos, emphasizing the significance of robust manipulation recognition methods. While much attention has been given to face manipulation detection in still images, the identification of tampered faces in videos has received comparatively less focus. The authors highlight the effectiveness of recurrent convolutional models, a class of DL techniques that can exploit temporal information across image streams. They investigate the most effective ways to combine domain-specific face pre-processing approaches with variations in these designs by a great deal of experimentation. Targeting Deepfake, Face2Face, and FaceSwap altered faces in particular, their method achieves superior results on video-based facial alteration benchmarks. The calculation was approved out on FaceForensics++ dataset, which resulted in significant improvements over previous state-of-the-art methods.

A W-Net architecture-based method for identifying and pinpointing areas of video that have been altered through the use of copy-move forging method is put forth by Tokas et al. [15]. Their approach demonstrates high efficiency in detecting forged videos, even in complex scenarios involving dynamic backgrounds or intricate object movements. Utilizing temporal copy and paste (TCP) video in-painting approaches, a model is able to recognize regions of the video frame that have been altered. Through the use of lossless video clips, the authors achieve optimal results, showcasing the model's robustness. The developed algorithm simplifies the task by requiring only a video clip as input, offering a streamlined solution for video forgery detection.

Rimsha Rafique et al. [16] propose an automated method for classifying deepfake images by combining DL and ML techniques. They identify the restrictions of existing ML systems that rely on handcrafted feature extraction, which often fail to capture complex patterns and generalize poorly to unseen data. To overcome this issues, the suggested framework first conducts Error Level Analysis (ELA) to detect potential image modifications. CNNs are then used to identify deep features from the altered image. SVM and K-NN are used to classify the extracted feature vectors, with hyper-parameter adjustment to improve efficiency. The suggested method, utilizing Residual Networks and K-Nearest Neighbors, achieved the highest accuracy, demonstrating the technique's efficiency and robustness. This approach shows promise for deepfake image detection and mitigating the risks associated with misinformation, slander, and propaganda.

A motion residual and DL-based forensic technique is presented by Raj et al. [17] to identify object-based fabricated frames in security footage. They emphasize the vulnerability of recorded video footage to manipulation, where objects can be easily removed using low-cost and readily accessible video editing tools. To ensure the integrity of sensitive surveillance videos, the authors propose a verification process before such videos are accepted as evidence. The approach begins by calculating motion residuals from video sequences, which capture forgery footprints resulting from manipulation. These motion residuals are then provide for into a VGG-16 network to classify frames as authentic or forged. This method offers a reliable solution for identifying tampered frames in surveillance footage, helping maintain the authenticity of video evidence.

Lakshmi et al.[18] propose a DL-based approach for detecting forged content in videos, addressing the growing concern of video tampering facilitated by the rise of digital technology. As digital videos play a crucial role in various aspects of life, ensuring their authenticity is vital. The suggested method classifies videos as authentic or tampered by employing two key techniques: patch analysis and error level analysis. In the patch model, the video frames are pre-processed and divided into patches for detailed examination. For error level analysis, the frames undergo compression, and differences are computed to detect inconsistencies. The ResNet50 DL classifier is applied to both methods to effectively detect video forgery. By combining these techniques, the suggested approach provides a reliable means of identifying tampered videos.

According to Sohaib et al. [19], there are increasing concerns about the negative applications of DL, especially in the production of deepfakes that is utilized to manufacture faces in media like pictures and movies. While DL has many positive uses in fields like medicine, animation, and cybersecurity, it has also led to issues like defamation, blackmail, and privacy violations. This work suggests a forgery detection system that utilizes a CNN-LSTM model. The process begins by extracting faces from video frames using MTCNN, followed by spatial feature extraction through a pre-trained Xception network. After that, LSTM) networks are used to extract temporal information. Lastly, classification is done to find out if the video is authentic or not. When evaluated on Google Deepfake AI dataset, the system has demonstrated an excellent accuracy in distinguishing between real and fake films, as well as the capacity to recognize low-quality videos.

Ding et al. [20] utilize deep transfer learning for face swapping detection, achieving true positive rates with minimal false alarms. Unlike existing methods that focus solely on detection accuracy, their approach also provides uncertainty estimates for each prediction, which is essential for ensuring trust in the deployment of such detection systems. In addition, the authors compare their method's performance with human subjects by building a website to collect pairwise comparisons of images. Based on these comparisons, they derive a consensus ranking from the most real to the most fake images.

Despite achieving high accuracy, existing forgery detection methods face challenges such as high computational complexity, reliance on extensive labeled datasets, and vulnerability to adversarial attacks. Many approaches struggle with generalization across diverse manipulation techniques and suffer from performance degradation in low-quality or compressed videos. To address these limitations, the proposed work aims to develop a more efficient and robust DL framework that enhances detection accuracy while reducing computational overhead, ensuring better adaptability to real-world forgery detection scenarios.

PROPOSED METHODOLOGY

The proposed methodology for video forgery detection is a systematic process designed to analyze video content and determine its authenticity. The workflow consists of multiple stages, each addressing limitations in traditional approaches using the proposed techniques for improved performance. The process begins with segmenting the input video into individual frames to facilitate detailed examination. Traditional frame extraction methods often suffer from inefficiencies that can lead to inconsistencies in analysis and loss of temporal information. To overcome these limitations, the proposed methodology employs an optimized frame extraction mechanism. This mechanism ensures uniform sampling, preserving temporal integrity of the video content and providing a robust foundation for subsequent steps. Once the frames are extracted, they undergo pre-processing to enhance their quality and reliability. Conventional pre-processing methods, which are often limited to noise removal or contrast adjustments, may fail to address diverse video quality issues effectively. The proposed system introduces advanced pre-processing techniques, including adaptive noise filtering, dynamic resizing of frames to a standardized dimension and contrast enhancement. These enhancements ensure consistency across frames and mitigate quality variations, which are crucial for accurate forgery detection. Feature extraction plays a vital role in identifying indicators of tampering, such as spatial inconsistencies, texture anomalies, and compression artifacts. Traditional feature extraction methods may struggle to detect subtle manipulations, reducing their effectiveness. To address this, the proposed system combines handcrafted and DL-based features. Handcrafted features, includes histogram analysis and texture patterns, and are extracted. This approach ensures the detection of both low-level and high-level forgery cues, significantly improving detection accuracy. The extracted features are send itto a trained detection module for classification. Traditional methods often face challenges such as high computational costs and memory constraints, particularly deal with large datasets. The proposed methodology mitigate these issues by proposing lightweight robust models optimized for efficient processing. Both SVM and CNN are used for training and classification, leveraging their respective strengths

in handling linear and non-linear patterns in the data. The detection module is trained on extensive labeled datasets of authentic and tampered videos, ensuring high reliability in classifying frames as real or forged. By systematically addressing the limitations of traditional methods and integrating advanced techniques at each stage, the proposed methodology provides a robust and efficient framework for video forgery detection.

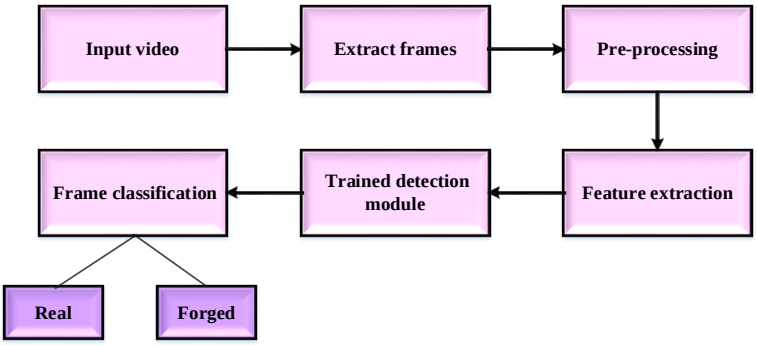


Figure 1 Work flow diagram for the proposed work

Pre-processing techniques

The proposed methodology begins by extracting individual frames from the input video, which are then subjected to a pre-processing stage to ensure consistency and quality. The extracted frames are analyzed in a sequential manner to capture both spatial and temporal patterns essential for forgery detection. This method facilitates a detailed analysis of the video's content while preserving its inherent temporal dynamics. To enhance the reliability of the analysis, pre-processing techniques are employed. These include resizing, which standardizes the dimensions of frames to a uniform size, ensuring compatibility with subsequent processing stages. For this methodology, the frames are resized to a fixed dimension (128 × 128 pixels), optimizing computational efficiency while retaining critical visual details. Gray-scale conversion is another critical step, which simplifies the color information in frames by reducing them to a single intensity channel. This not only reduces computational complexity but also highlights key spatial patterns and anomalies that may indicate tampering. Finally, normalization is applied to the frames, scaling the pixel intensity values to a standardized range. This step mitigates variations in lighting and contrast across frames, ensuring a consistent representation of features. By eliminating inconsistencies, normalization improves the accuracy of subsequent feature extraction and classification process. Together, these pre-processing techniques enhance the quality and uniformity of the extracted frames, enabling the proposed system to effectively capture spatial and temporal patterns crucial for detecting video forgery.

Process of SVM for video forgery detection

SVM is a robust ML classifier, widely recognized for its efficacy in binary classification tasks. By consuming SVM, the proposed methodology aims to distinguish authentic video segments from tampered ones with high-precision. This process integrates feature extraction, frame differencing, and classification using a linear SVM kernel, ensuring computational efficiency and interpretability. The steps of the SVM based video forgery detection methodology are detailed below.

Feature extraction

The pre-processed frames are subjected to feature extraction to capture essential indicators of tampering. These indicators include texture anomalies, spatial inconsistencies, or disruptions in edges, all of which can reveal tampered regions. To achieve this, the proposed methodology employs a combination of handcrafted feature descriptors that excel in capturing various dimensions of forgery evidence:

- **Histogram of Oriented Gradients (HOG**): The HOG descriptor is widely recognized for its capability to capture shape and structural information from images by analyzing the distribution of gradient orientations. It works by separating an image into small connected regions called cells and evaluating the histogram of gradient directions with in each cell. The resulting histograms are normalized to enhance robustness against illumination changes. This method is particularly effective in identifying tampered regions where textures have been altered due to manipulation, such as smudging, cloning or reshaping. By focusing on edge and

contour patterns, HOG ensures that any deviations in the structural consistency of an image are highlighted, making it a reliable tool for detecting forgeries in both static and dynamic frames.

For an image $I\ (x, y)$, compute gradients:

$$G_x = I(x+1, y) - I(x-1, y)$$

$$G_y = I(x, y+1) - I(x, y-1)$$

where G_x and G_y represent the horizontal and vertical gradient, respectively.

The gradient magnitude is computed as:

$$M(x, y) = \sqrt{G_x^2 + G_y^2}$$

The gradient orientation is given by:

$$\theta(x, y) = tan^{-1}\left(\frac{G_y}{G_x}\right)$$

Each cell accumulates votes based on gradient magnitudes into orientation bins. The histogram is formulated as:

$$H_b = \sum_{(x,y)\in cell} M(x, y).\,\delta(\theta(x, y) - b)$$

where H_b is the histogram count for bin b, and δ is the Kronecker delta function.

Block normalization ensures illumination invariance:

$$V = \frac{H}{\|H\|_2 + \epsilon}$$

Where H is the concatenated histogram vector, $\|.\|_2$ is the L2 norm, and ϵ is a small constant to avoid division by zero.

- **Local Binary Patterns (LBP):** The LBP descriptor is a powerful texture operator that examines local pixel neighborhoods to identify micro-textural patterns. It assigns a binary value to each pixel by comparing its intensity with that of its neighboring pixels, creating a unique texture signature. This method is particularly sensitive to subtle anomalies, such as those caused by copy-move or splicing forgery. In copy-move forgeries, duplicated regions often introduce repetitive textural inconsistencies, while in splicing forgeries, the pasted region may differ in texture from the surrounding areas. LBP excels in capturing these inconsistencies, making it a critical feature for identifying forgery-related micro-textural changes.

For a given pixel I_c and its P surrounding neighbors at radius R, compute:

$$LBP_P^R = \sum_{p=0}^{P-1} s\left(I_p - I_c\right).2^p$$

where:

$$s(x) = \begin{cases} 1, & x \geq 0 \\ 0, & x < 0 \end{cases}$$

I_p represents the intensity of the p-th neighbor, I_c is the center pixel intensity.

The LBP feature vector is created by computing the histogram:

$$H(k) = \sum_{x,y} \delta(\,LBP_P^R(x, y) - k), \qquad k = 0,1,\ldots.,2^P - 1$$

where δ is the Kronecker delta.

By combining these feature descriptors, the proposed methodology leverages the strengths of each to ensure comprehensive forgery detection.

Using the HOG methodology described earlier, generate HOG feature vector H_{HOG}:

$$H_{HOG} = \{h_1, h_{2,\ldots,} h_n\}$$

where h_i are the normalized HOG descriptors.

Using the LBP method, extract the feature vector H_{LBP}:

$$H_{LBP} = \{l_1, l_{2,\ldots,} l_m\}$$

where l_j are the LBP texture descriptors.

The final hybrid feature vector is:

$$H_{Hybrid} = [H_{HOG} \parallel H_{LBP}]$$

where ‖ represents concatenation.

This hybrid approach enhances robustness by capturing both edge-based structural details and fine-grained texture patterns, making it highly effective for video forgery detection. HOG captures the global structural and shape anomalies, LBP focuses on localized textural inconsistencies in frames. Together, these descriptors provide a multi-faceted approach to identifying forgeries, improving the robustness and accuracy of detection process. This hybrid feature extraction strategy ensures that a wide range of forgery techniques, from spatial manipulations to temporal alternations, are effectively identified, making it a critical component of the proposed forgery framework.

Frame Differencing

To enhance feature representation, the Difference of Consecutive Frames (DOC) technique is employed to calculate pixel-level differences between consecutive frames. This approach is particularly effective in highlighting sudden changes in motion or texture, which may result from forgery. By aggregating these pixel-level differences into feature vectors, the DOCF based descriptor offers improved discrimination between authentic and tampered frames.

The DOCF technique is a temporal analysis method specifically designed to detect inconsistencies across consecutive frames in video sequences. By computing the pixel-wise difference between two consecutive frames, it isolates abrupt changes in texture or motion that may indicate forgery. Such anomalies are often caused by tampering operations, including the insertion or removal of objects or unnatural transitions due to frame interpolation or duplication. In tampered videos, these sudden pixel-level variations disrupt the temporal continuity, making them detectable through DOCF. A DOCF based descriptor is particularly valuable in scenarios where spatial descriptors fail to capture temporal inconsistencies. Deepfake generation or frame manipulation often alters the temporal flow of videos, creating inconsistencies that DOCF can effectively highlight. By focusing on these anomalies, the DOCF technique enables the detection of forged segments with greater accuracy, providing a robust means of distinguishing between authentic and tampered frames.

SVM model training

Once features are extracted, the SVM mode is trained to distinguish between authentic and forged frames. This process involves key steps such as inputting feature vectors, selecting a linear kernel, and optimizing the hyperplane.

- **Feature vector input**: The extracted feature vectors $X_i \in \mathbb{R}^n$ (where n is the dimensionality of feature space) represent the characteristics of each frame. Each feature vector is associated with a binary level $y_i \in \{0, 1\}$, where

$$y_i = 0: \quad \text{The frame is authentic.}$$
$$y_i = 1: \quad \text{The frame is forged.}$$

The training dataset consists of m feature-label pairs: $\{(x_{i,} y_i)\}_{i=1}^{m}$

- **Linear kernel selection:** A linear kernel is used for simplicity and efficiency, which implies that the decision boundary is a linear function of input features. The decision function is defined as $f(x) = w^T x + b$, where $w \in \mathbb{R}^n$ is weight vector, b $\in \mathbb{R}^n$ is the bias term. $f(x)$ is decision score, with frame classified as:

$$\text{Class}= \begin{cases} +1, \ if\ f(x) \geq 0 \\ -1, if\ f(x) < 0 \end{cases}$$

- **Hyperplane optimization:** In order to maximize the margin among two classes, the SVM looks for best hyperplane to divide them. The distance between hyperplane and closest data points—also referred to as support vectors—from each class is the margin. For a linear SVM, the optimization problem is stated as follows: $\underset{w,b}{min}\ \frac{1}{2}\|w\|^2$, subject to constraints:

$$y_i(w^T x_i + b) \geq 1, \quad \forall i \ \in \{1, \ldots\ldots, m\}$$

This ensures that all data points are correctly classified with a margin of at least $1/\|w\|$.

- **Handling Non-separable data**: To handle cases where the data is not perfectly linearly separable, slack variables $\xi i \geq 0$ are introduced. The optimization problem becomes:

$$\underset{w,b,\xi}{min}\ \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{m} \xi_i$$

subject to: $y_i(w^T x_i + b) \geq 1 - \xi_i, \quad \xi_i \geq 0, \quad \forall i$

Here, $C > 0$ is the regularization parameter that controls trade-off between maximizing the margin and minimizing classification errors.

- **Solving the optimization problem:** The dual formulation of SVM optimization is solved to determine the weight vector w and bias b. The dual problem is expressed as:

$$\underset{\alpha}{max} \sum_{i=1}^{m} \alpha_i - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} \alpha_i\, \alpha_j y_i y_j x_i x_j$$

subject to:

$$0 \leq \alpha_i \leq C, \sum_{i=1}^{m} \alpha_i y_i = 0$$

where α_i are the Lagrange multipliers. The weight vector is then computed as:

$$w = \sum_{i=1}^{m} \alpha_i\, y_i x_i$$

The bias term b is determined using support vectors:

$$b = y_i - w^T x_i, \text{ for any support vector } i.$$

- **Decision function:** After training, the decision function for classifying a new frame with feature vector x is:

$$f(x) = \sum_{i=1}^{m} \alpha_i\, y_i {x_i}^T x + b$$

If $f(x) \geq 0$), the frame is classified as authentic; otherwise it is classified as forged.

Testing and evaluation

After training, SVM model undergoes testing to evaluate its performance. The model classifies video segments as authentic or forged by analyzing the feature vector of frames. Metrics such as accuracy, precision, recall, f1-score are used to access the system's robustness and reliability. Each segment is assigned a binary label (authentic and forged), enabling comprehensive evaluation of system's classification capabilities.

Forgery localization

When a video segment is classified as forged, additional post-processing steps are employed to accurately identify the specific frames and regions within those frames that were tampered with. This process begins with a detailed analysis of feature vector distribution across consecutive frames. By examining these distributions, the system identifies patterns or inconsistencies that deviate from normal, authentic behavior. To localize tampered areas, spatial anomaly detection techniques are applied. Regions within the frame that exhibit abnormal texture or edge patterns, such as abrupt changes in gradient orientation or micro-textural inconsistencies are flagged as potential forgery hotspots. Unnatural transitions caused by object insertion, removal, or duplication can be highlighted using HOG and LBP, which capture texture and edge irregularities effectively. This precise localization of tampered regions not only aids in detecting forged content but also provides valuable insights int the nature and extent of manipulation, enhancing the robustness of forgery detection process. Table 1 explain about the Algorithm for SVM based video forgery detection.

Table 1 Video forgery detection using SVM

Input:
- Video datasets (manipulated and original)
- Frame rate
- Labels

Output:
- Performance metrics: Accuracy, Precision, Recall, and F1 Score

Step 1: Data Preprocessing

1. **Load Video Data:** Load video files along with their corresponding labels.
2. **Frame Extraction:** Extract frames from each video using the formula:

$$Frame = Video[t \times Frame - Rate]$$

where t is the time index.

3. **Frame Resizing:** Resize each extracted frame to a fixed dimension (128×128 pixels).
4. **Feature Normalization:** Flatten each frame into a feature vector and normalize pixel values using:

$$X_{norm} = \frac{X}{255}$$

Step 2: Feature Extraction

1. **Extract Feature Vectors:** For each video, extract feature vectors f_i where i represents the frame index.
2. **Feature Representation:** Example features include pixel intensities and statistical properties relevant for forgery detection.

Step 3: Train-Test Split

1. **Partition Dataset:** Split the dataset Dinto training D_{train} and testing D_{test} subsets, ensuring no overlap:

$$D = D_{train} \cup D_{test}, D_{train} \cap D_{test} = \emptyset$$

Step 4: Train the SVM Model

1. **Define the SVM Optimization Problem:** The objective is to decrease the following function:

$$\frac{1}{2}\|w\|^2 + C\sum_{i=1}^{m}\xi_i$$

Subject to: $y_i(w^T x_i + b) \geq 1 - \xi_i, \quad \xi_i \geq 0$

where w^T is weight vector, b is bias, ξ_i are slack variables.

2. **Model Training:** Train the SVM classifier using the training dataset D_{train}.

Step 5: Testing and Prediction

1. **Label Prediction:** For each test sample x, predict the label $\hat{y}$ using:

$$\hat{y} = sign\ (w.x + b)$$

Step 6: Performance Evaluation

1. **Calculate Evaluation Metrics:** Compute the classification performance using:

$$Accuracy = \frac{Number\ of\ correct\ predictions}{Total\ number\ of\ predictions}$$

$$Precision = \frac{TP}{TP+FP}$$

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

$$F1\ score = 2 \times \frac{Precision\ \times\ Recall}{Precision\ +Recall}$$

Return:
Performance metrics and confusion matrix.

The proposed SVM-based methodology combines the interpretability of handcrafted feature extraction with the computational efficiency of a linear kernel, offering effective solution for video forgery detection. Its computational efficiency, enabled by the lightweight linear kernel, makes it highly suitable for resource-constrained environments, ensuring fast and reliable detection without imposing significant overhead. The simplicity of the approach lies in its use of handcrafted features and linear classification, which ensures interpretability and ease of implementation. Despite its simplicity, the methodology is highly effective, as the integration of HOG, LBP, and DOCF descriptors allows for robust detection of tampered segments, even in challenging scenarios. Additionally, the system is scalable, capable of handling larger datasets or being extended with additional feature descriptors to enhance performance further. By leveraging handcrafted features and a linear SVM classifier, this methodology strikes a balance between efficiency, effectiveness, and simplicity, creating it an ideal choice for preliminary or low-resource video forgery detection tasks.

Process of CNN for video forgery detection

CNN have been frequently applied in image classification applications due to their capacity to learn spatial elements well. In the area of video forgery recognition, CNN is essential for spotting subtle irregularities brought on by manipulation. By using convolutional layers for feature extraction and pooling layers for dimensionality reduction, CNNs can detect forged content with high accuracy. Unlike traditional methods that require extensive manual feature engineering, CNNs automatically capture textural inconsistencies and visual artifacts across video frames, making then highly effective for forgery detection.

CNN architecture

A CNN processes image data using multiple layers, containing convolutional, activation, pooling and fully connected layers. The architecture for a forgery detection CNN consists of:

- Input layer: Accepts video frames as input, usually resized and pre-processed.
- Convolutional layers: Extracts spatial features using multiple filters.
- Activation function (ReLU): Introduces non-linearity to improve learning.
- Pooling layers (Max pooling): Reduces spatial dimensions to retain crucial data.
- Fully connected layers: Flatten feature maps and classifies frame as forged or authentic.
- Output layer: generates the final forgery prediction.

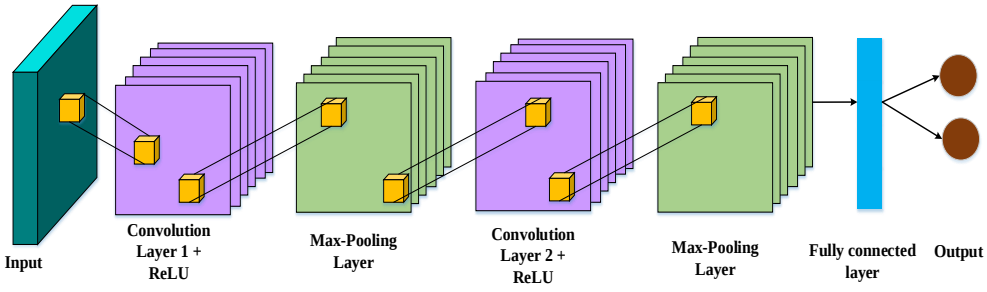


Figure 2 CNN architecture for the proposed work

Convolutional Layer: The convolution operation between an input image I and a filter K is described as:

$$(I * K)(x, y) = \sum_{m} \sum_{n} I(x + m, y + n) \cdot K(m, n)$$

where m, n are the dimensions of the filter. This operation extracts meaningful patterns from the input data, generating feature maps that highlight important spatial characteristics.

Activation function: An activation function is used to add non-linearity following the convolution process. ReLU is commonly used:

$$f(x) = \max(0, x)$$

Pooling layer: Pooling layers preserve crucial data while reducing the spatial dimensions of feature maps. The operation for max pooling is provided by:

$$P(x, y) = max\{I(x + m, y + n) \mid m, n \in \ pool\,size\}$$

This layer enhances the computational efficiency and reduces overfitting.

Fully connected layer: Flattened feature maps are passed through fully connected layers to perform classification. The mathematical representation is expressed as:

$$\boldsymbol{z = w^T x + b}$$

where w and b are the weights and bias, respectively, and x is the input vector.

Loss function: For binary classification in forgery detection, binary cross-entropy loss function is utilized:

$$\boldsymbol{\mathcal{L} = -\frac{1}{N}\sum_{i=1}^{N} [y_i \log (p_i) + (1 - y_i) \log (1 - p_i)]}$$

where y_i is ground table label, and p_i is predicted probability for sample i.

Backpropagation and weight updates: To optimize the model, gradients of loss function with respect to weights are calculated using chain rule and updated via gradient descent:

$$\boldsymbol{w_{t+1} = w_t - \eta \frac{\partial \mathcal{L}}{\partial w_t}}$$

where η is learning rate, controlling the step size of updates.

CNNs effectively detect video forgery by leveraging convolutional layers for feature extraction, activation functions for non-linearity, and pooling layers for dimensionality reduction. Fully connected layers classify forgery, optimized through back-propagation and gradient descent. This approach ensures high accuracy, robustness, and adaptability across various video formats and forgery techniques. Table 2 explains about the algorithm for detecting forgery using CNN technique.

Table 2: CNN- based Video Forgery detection

Input:
- Video datasets (manipulated and original)
- Frame rate
- Labels

Output:
- Classification performance metrics (Accuracy, Precision, Recall, F1 Score)

Step 1: Frame Extraction

1. Load video files and their corresponding labels.
2. Extract frames at regular intervals using the formula:

$$Frame = Video[t \times Frame - Rate]$$

where t is the time index.

3. Resize each extracted frame to a fixed dimension (128×128 pixels).
4. Flatten each frame into a feature vector and normalize pixel values using:

$$X_{norm} = \frac{X}{255}$$

Step 2: Dataset Preparation

1. Stack frames into a tensor

$$\mathbf{X} \in \mathbb{R}^{B \times T \times H \times W \times C}$$

where B is batch size, T is number of frames, H is height, W is width, and C is the number of channels.

2. Assign binary labels to videos

$y = 1$ for manipulated, $y = 0$ for original

3. Split data into training, and test sets.

Step 3: Define CNN Architecture

1. Define the CNN layers:
   - Convolutional Layer: $Z = ReLU\ (w * x + b)$

where w is the filter kernel, $*$ denotes convolution, and b is the bias term.

- Pooling Layer: $P = MaxPool\ (Z, pool\_size)$

2. Add fully connected layers and the output layer with sigmoid activation:

$$\hat{y} = \sigma(W_{fc} \cdot X_{fc} + b_{fc})$$

Step 4: Model Training

1. Define the loss function using binary cross-entropy:

$$\mathcal{L} = -\frac{1}{N}\sum_{i=1}^{N} [y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)]$$

2. Train CNN on the training set.

Step 5: Testing and Prediction

1. Predict labels for test data using:

$$\hat{y} = sigmoid\ (f(x))$$

Step 6: Evaluate Performance

2. Calculate evaluation metrics: Compute the classification performance using:

$$Accuracy = \frac{Number\ of\ correct\ predictions}{Total\ number\ of\ predictions}$$

$$Precision = \frac{TP}{TP+FP}$$

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

$$F1\ score = 2 \times \frac{Precision\ \times\ Recall}{Precision\ +Recall}$$

Return:

1. Performance metrics and confusion matrix.

Key advantages of CNNs in forgery detection

Automated Feature Extraction: CNNs automatically learn significant patterns from data, making them powerful feature extractors. In video forgery detection, they identify intricate details such as inconsistencies in object edges, unnatural textures, and visual artifacts introduced during splicing or copy-move manipulations. These subtle variations are often difficult to detect using traditional techniques.

Temporal and Spatial Analysis: Since video frames exhibit both spatial and temporal coherence, CNNs analyze not only individual frame anomalies but also inconsistencies across consecutive frames. Spatial features such as texture deviations and lighting discrepancies caused by tampering are effectively captured. Additionally, CNNs are often integrated with 3D convolutions to process sequential data, enhancing their ability to detect forgery in dynamic scenes.

Network Architecture and Training: Multiple convolutional and pooling layers, followed by fully connected layers for classification, comprise a standard CNN model for video forgery recognition. A dataset comprising both real and fake frames is used to train the network. By minimizing classification errors, the CNN learns to distinguish between authentic and manipulated content. The model is exposed to various forgery techniques, such as object splicing and copy-pasting, enabling it to generalize effectively to different types of forgeries.

Detection and Localization: Once trained, the CNN can analyze video frames to detect potential forgeries. In addition to classification, CNNs can localize forged regions within frames. By examining feature maps in convolutional layers, the network highlights specific areas that exhibit signs of tampering, allowing for precise localization of forged regions.

Performance and Robustness: CNN-based forgery detection models can be fine-tuned to improve robustness across different video qualities, resolutions, and compression formats. This adaptability is particularly beneficial for detecting forgeries in social media videos and compressed surveillance footage, where artifacts may obscure tampered content. Cross-dataset validation further enhances the model's reliability by estimating its performance on diverse datasets**.**

The proposed forgery detection system offers an automated, high-accuracy solution that eliminates the essential for manual feature engineering. It effectively identifies tampered regions, even in compressed or low-quality videos. This method enhances security in digital media authentication, making it highly beneficial for deepfake detection applications.

DATASET AND EXPERIMENTAL DETAILS

A thorough description of the datasets used and the specifics of the implementation tools are given in this section.

Dataset Description

The dataset utilized in this study includes 101 videos, including both manipulated and original content. The altered videos were artificially generated using fundamental forgery techniques, specifically copy-move and splicing, ensuring a diverse representation of tampering cases. To develop and evaluate the video forgery detection model, two major dataset configurations were considered, each with different proportions of training and testing videos. The dataset was splitted into training, validation, and testing sets in a 70:20:10 ratio, ensuring that the model had sufficient data for evaluation.

In the first configuration, a smaller subset of 33 videos was utilized, where 22 videos were assigned for training and 11 for testing. The second configuration adopted a larger subset of 57 videos, with 42 videos used for training and 15 for testing. The dataset maintained a balanced distribution of altered and original videos in both training and testing sets, ensuring fairness. In the first configuration, the training set contained 17 tampered and 5 benign videos, while the testing set included 6 tampered and 5 benign videos. This balanced representation enable the model to learn forgery-specific features while preventing overfitting, thereby improving its generalization capability.

Implementation tools

The experiments were conducted on a high-performance system equipped with an Intel Core i7 10th generation processor, 16GB of RAM, and a 1TB SSD. This robust hardware configuration efficiently handled computationally intensive tasks such as video preprocessing and model training. The software environment was based on Python 3.8, incorporating essential libraries and frameworks. DL models were implemented using TensorFlow and Keras, while OpenCV facilitated video frame extraction and pre-processing. Traditional ML techniques (SVM), were applied using Scikit-learn. Additionally, Matplotlib and Seaborn were employed for visualizing performance metrics, and NumPy was utilized for optimized numerical computations. This combination of hardware and software ensured a seamless and efficient execution of the proposed methodology.

Dataset Splits and evaluation

Table 3: Dataset splitting for training and testing

Set	Number of Video	Train	Test
First Set	33	22	11
Second Set	57	42	15

Table 3 offers the splitting of training and testing datasets and the comparison for dataset is provided in figure 3. Figure 3 presents by comparing the number of videos allocated for training and testing in two different dataset configurations. The x-axis signifies the dataset type (train and test), while the y-axis indicates the number of videos. The first set (blue) consists of 33 videos, with 22 utilized for training and 11 for testing. The second set (yellow) include 57 videos, with 42 allocated for training and 15 for testing.

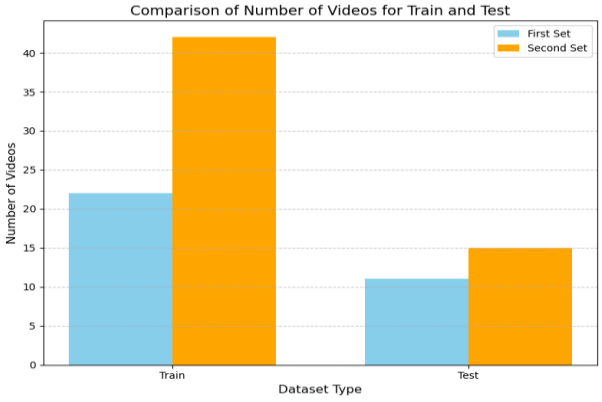


Figure 3: Splitting of dataset for training and testing

Table 4: Distribution of manipulated and original videos in 1st dataset

Set	Total	Manipulated	Original
Train	22	17	5
Test	11	6	5

Table 4 presents the distribution of manipulated and original videos in the first dataset configuration, which consists of a total of 33 videos. The graphical representation of the table is illustrated in figure 4. The x-axis signifies video categories (manipulated and original), while y-axis indicates the number of videos. The blue bars signify training videos, and the yellow bars denote testing videos. The training set consists of 22 videos, of which 17 are manipulated, and 5 are original. Similarly, the testing set contains 11 videos, with 6 being manipulated and 5 being original. This balanced distribution ensures that the model is exposed to a sufficient number of manipulated and original videos during both training and testing phases. The higher proportion of manipulated videos in the training set helps the model learn tampering patterns effectively while maintaining a fair representation of original content to avoid bias.

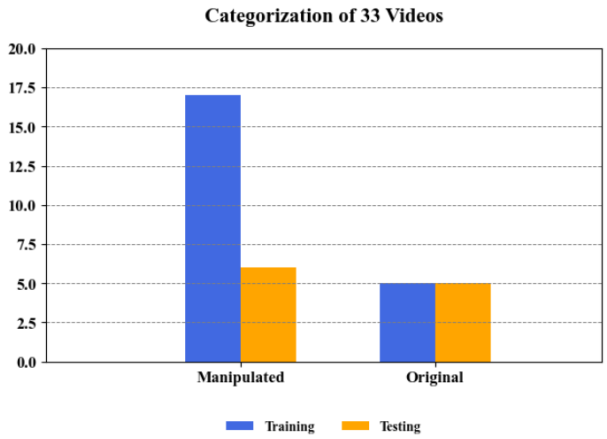


Figure 4: Categorization of 33 videos

Table 5: Distribution of manipulated and original videos in 2nd dataset

Set	Total	Manipulated	Original
Train	42	25	17
Test	15	7	8

Table 5 presents the distribution of manipulated and original videos in the second dataset configuration, which consists of a total of 57 videos. Figure 5 illustrates the categorization of 57 videos in the second dataset, distinguishing between manipulated and original content for both training and testing phases. The training set consist of 42 videos, with 25 manipulated and 17 original videos, whereas testing set contains 15 videos, with 7 manipulated and 8 original videos. The balanced representation ensures that the model learns distinguishing features of tampered and authentic videos, reducing overfitting risk and improving generalization during evaluation.

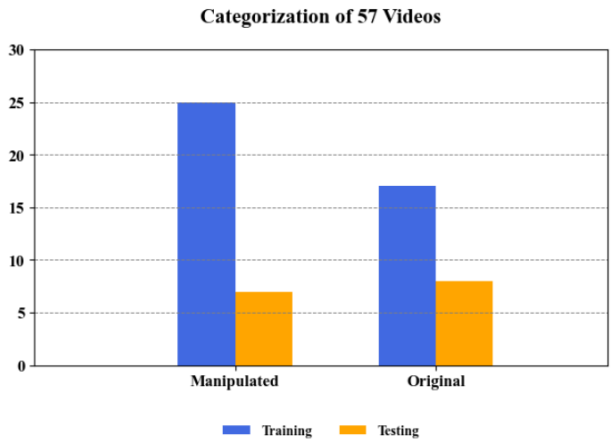


Figure 5: Categorization of 57 videos

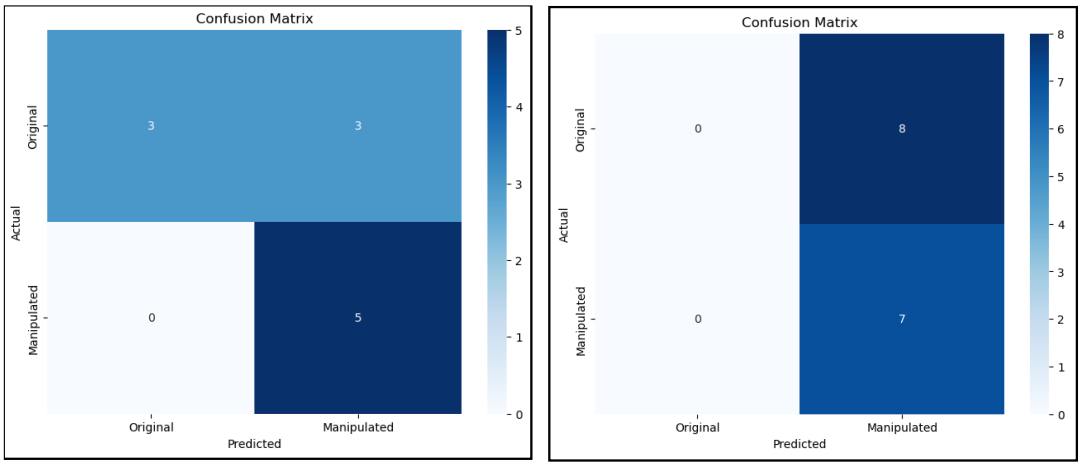
To ensure a comprehensive evaluation, multiple dataset splits were employed, allowing the detection model to be assesses across different sample sizes, ranging from higher to lower data availability. This variation was crucial for analyzing model's ability to generalize effectively, as it tested performance under varying volumes of training data. In all configurations, the training set was utilized for tuning model parameters, while testing set provided an independent assessment of accuracy and reliability. This structured division ensured that the performance metrics accurately reflected the model's effectiveness in real-world scenarios, where video quality, format and tampering methods may differ.

RESULTS AND DISCUSSION

Evaluation results for SVM

The evaluation of SVM for video tagging tasks provided valuable insights into its performance across datasets of different sizes, specifically 33 and 57 videos. The analysis considered key metrics includes accuracy, precision, F1 score, and ROC AUC, highlighting the impact of data quantity on classifier effectiveness.

Figure 6 shows the confusion matrix for video forgery detection using SVM technique on datasets comprising 33 and 57 videos. These matrix illustrate the classifier's performance in distinguishing between original and manipulated videos by comparing actual and predicted labels.



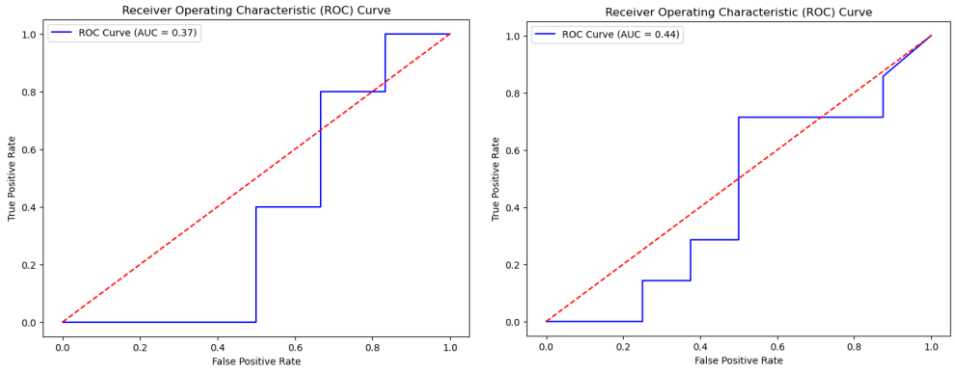
(a) 33 Videos **(b) 57 Videos**

Figure. 6. Confusion matrix for the proposed work

The confusion matrices reveal that the SVM classifier performed well in detecting manipulated videos but struggled with original videos, particularly as the dataset size increased. In the smaller dataset (33 videos), it correctly classified all manipulated videos but misclassified some original ones. However, in the larger dataset (57 videos), it exhibited a strong bias toward predicting manipulated videos, misclassifying all original videos. This indicates that the

classifier's generalization ability declined with a larger dataset, likely due to overfitting or data imbalance. Further optimization through feature selection, kernel tuning, or alternative models is needed to enhance accuracy.

Figure 7 illustrates the ROC performance of the SVM classifier on datasets with 33 and 57 videos. The ROC curve plots the TP rate against FP rate, providing insight into the classifier's ability to distinguish between original and manipulated videos. The ROC curves for 33 and 57 videos show poor classification performance, with AUC values of 0.37 and 0.44, correspondingly. Both curve remains close to the diagonal, indicating the SVM struggles to distinguish original from manipulated videos. The low AUC values highlight the need for improved feature selection, kernel tuning, or alternative classification methods.



(a) 33 Videos **(b) 57 videos**

Figure 7 ROC curve for SVM technique

Figure 8 demonstrates training and testing accuracy of an SVM model over 25 epochs. Initially, both accuracies start at approximately 47% and gradually increase. Training accuracy rises rapidly, stabilizing between 60% and 62.5%, while testing accuracy improves more slowly, fluctuating before reaching nearly 60% at epoch 25. The noticeable gap between training and testing accuracy suggests potential overfitting, where the method achieves well on training data.

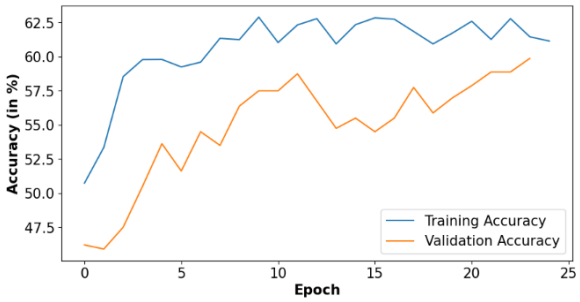


Figure 8: Training and validation Accuracy

Table 6. Results obtained using SVM technique

SVM				
Total Video	**Accuracy**	**Precision**	**F1-score**	**ROC AUC**
33	72.73%	62%	77%	37
57	46.67%	47%	64%	44

Table 6 shows the overall results obtained for SVM technique. In terms of accuracy, SVM achieved 46.67% on the larger datasets (57 videos) but significantly improved to 72.73% on the smaller dataset (33 videos). This discrepancy suggested that the classifier struggled to generalize effectively when dealing with a larger and more diverse dataset, potentially due to under fitting or class imbalance. Conversely, its improved performance on the smaller dataset indicated that the model benefited from a more consistent and less complex distribution of features.

Precision values followed a similar trend, with the classifier achieving 0.47 for the larger dataset and 0.62 for the smaller one. This increase implied a moderate improvement in model's ability to correctly identify real positives when dealing with a less diverse dataset. The reduction in false positives in the smaller dataset contributed to this improvement, reinforcing the notion that a more homogeneous data distribution positively impacted precision. The F1-score, balances precision and recall, also showed an increasing from 0.64 in the larger dataset to 0.77 in the smaller one. This progression reflected the classifier's enhanced ability to correctly classify true positives while minimizing false negatives, particularly when working with a dataset of reduced complexity. ROC AUC values, which measures model's ability to differentiate between classes, remained low across both datasets, with scores of 0.44 and 0.37 for larger and smaller datasets, correspondingly. These results indicated that the decision boundary established by the SVM lacked robustness, suggesting overlapping feature distributions or limitations in the SVM kernel's ability to capture underlying patterns effectively.

Figure 9 presents the classification performance of the SVM for video forgery detection across two datasets, 57 and 33 videos. As mentioned in table 6, the values for each metric have been evaluated.

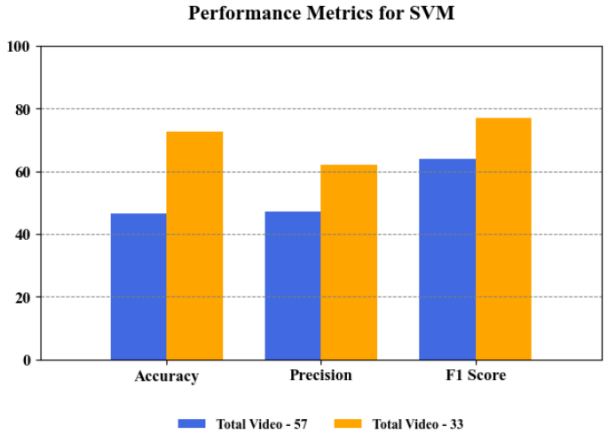
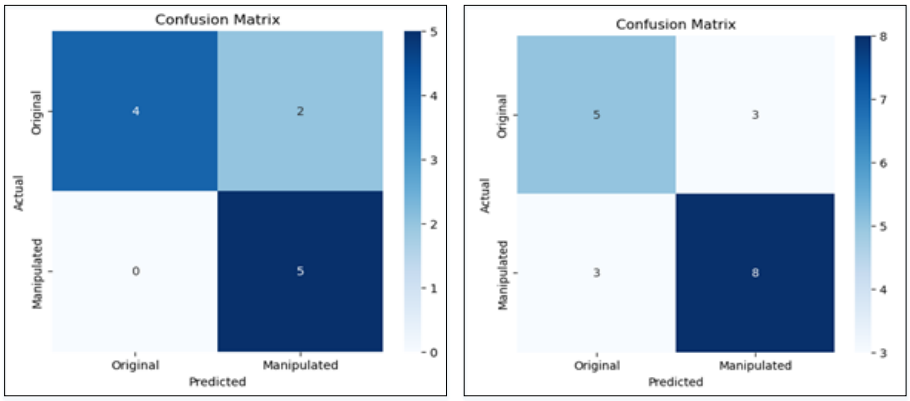


Figure 9: Overall performance metrics for SVM

The evaluation of SVM for video annotation and forgery detection emphasized the importance of pre-processing and feature engineering. Performance degradation on the larger dataset suggested that class imbalance or noise adversely affected the classifier's learning process. The low ROC AUC values indicated the need for further optimization of the SVM kernel or the exploration of alternative classifiers better suited to the dataset. For video forgery detection, SVM demonstrated performance variations across different dataset sizes. While achieving better accuracy and precision on the smaller dataset, its generalization ability declined with the larger dataset, leading to increased false positives. The low AUC values further highlighted the method's limited ability to differentiate between original and manipulated videos. These findings suggest that SVM may not be the most effective approach for video forgery detection, underscoring the need for advanced classifiers to improve detection accuracy and robustness.

Evaluation results for CNN

The evaluation of CNN classifier across datasets of varying sizes (33 and 57 videos) demonstrates its potential for video categorization. Figure 10 shows the confusion matrix for video forgery detection using CNN technique on datasets comprising 33 and 57 videos. These matrix illustrate the classifier's performance in distinguishing between original and manipulated videos by comparing actual and predicted labels.



(a) 33 videos **(b) 57 videos**

Figure 10: Confusion matrix for CNN

In 33 videos confusion matrix, the model correctly classified 4 original images but misclassified 2 as manipulated, while it perfectly identified all 5 manipulated images without any false positives. This indicates that the model was more confident in identifying manipulated images but showed some difficulty in distinguishing original ones. In 57 videos confusion matrix, the model improve its ability to recognize original images, correctly classifying five of them. However, 3 original images were misclassified as manipulated, and 3 manipulated images were incorrectly labeled as original. Despite this, the model still showed strong performance with eight correct predictions for manipulated images, demonstrating an improvement in overall classification but with a slight increase in FP.

Figure 11 illustrates training and validation loss, along with accuracy trends over multiple epochs. The training and validation loss graph shows a steep decline in the early epochs, indicating that the CNN quickly learned significant features, after the initial drop, loss values stabilized, suggesting that the design reached convergence without excessive overfitting. The similarity between training and validation loss patterns proposes that the technique generalizes well without significant variance. The training and validation accuracy graph highlights an increasing trend in training accuracy over epochs, demonstrating that the model effectively learned training data. Overall, CNN model exhibited strong classification performance, with a well-converged training process.

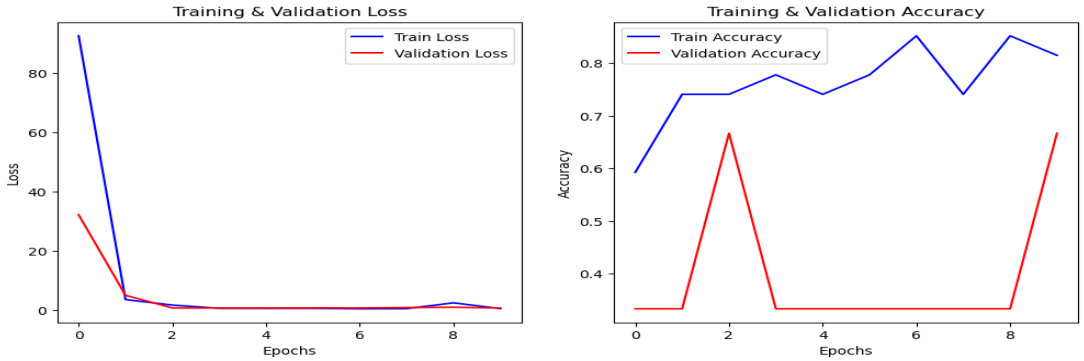


Figure 11. Training-Validation Loss and Accuracy for CNN

ROC curve presented in figure 12 illustrates the performance of CNN model in distinguishing between original and manipulated images. The ROC curve plots TPR against FPR at numerous threshold settings, providing insight into the model's discriminative ability. The blue curve signifies the actual performance of classifier, while diagonal dashed line signifies a random classifier with no predictive power. The AUC value is reported as 0.81, representing that the technique has a good ability to distinguish between two classes. An AUC value is reported as 0.81, indicating that the technique has a good ability to distinguish among two classes. Overall, ROC curve and AUC value demonstrate that the CNN model effectively differentiates between original and manipulated images.

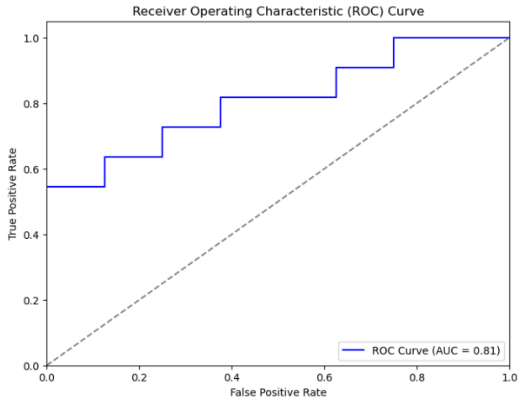


Figure 12: ROC curve for CNN technique

Table 7: Results obtained using CNN technique

CNN			
Total Video	**Accuracy**	**Precision**	**F1 Score**
57	73.33	0.64	0.78
33	81.82	0.71	0.83

Table 7 presents the overall results obtained from CNN technique. The CNN attained an accuracy of 73.33% on a larger dataset of 57 videos and improved to 81.82% on a smaller dataset of 33 videos. This indicates model's capability to capture meaningful patterns regardless of dataset size, with fewer videos allowing it to focus on more representative features. Precision values of 0.64 and 0.71, respectively, reflect the model's consistency in suitably recognizing positive cases while minimizing false alarms. The smaller dataset, with less diversity or noise, appears to provide an environment where the model performs better. The F1 score that balances precision and recall, increased from 0.78 to 0.83 when moving from the larger to the smaller dataset. This improvement highlights the CNN's strong performance, particularly in minimizing false negatives. The higher F1 score underscores the model's ability to attain a good balance between precision and recall under simpler conditions. Figure 12 depicts the overall performance graph for two video datasets using CNN model.

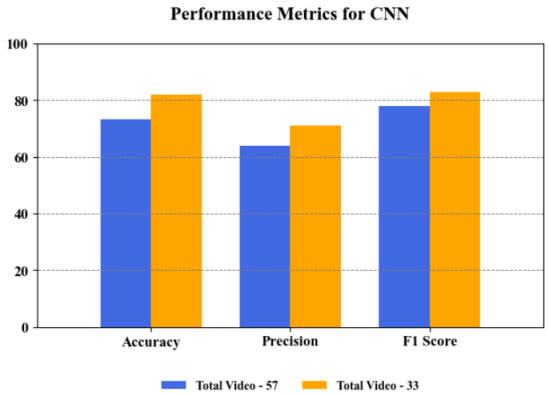


Figure 12: Overall performance metrics for CNN

The comparative evaluation of CNN and SVM for video categorization highlights key differences in their performance across several metrics, including accuracy, precision, and F1 score. The analysis was conducted on two datasets of varying sizes (57 and 33 videos), providing insights into their effectiveness. CNN demonstrated superior accuracy on both datasets, achieving 73.33% accuracy on the larger dataset and improving to 81.82% on the smaller dataset. In contrast, SVM recorded 46.67% and 72.73% accuracy, respectively, indicating CNN's ability to capture spatial and temporal features effectively. Precision values further reinforced CNN's dominance, with scores of 0.64 and 0.71 compared to SVM's 0.47 and 0.62. In terms of F1 score, CNN consistently outperformed SVM, achieving 0.78 on the

larger dataset and 0.83 on the smaller one, whereas SVM attained 0.64 and 0.77, respectively. This highlights CNN's robustness in handling imbalanced datasets and minimizing false negatives. Additionally, SVM exhibited low Area under the Curve (AUC) values of 0.44 and 0.37, indicating its limited capability in distinguishing between classes.

Overall, CNN proved to be more effective for video categorization, offering higher classification reliability and better generalization across datasets of different sizes.

DISCUSSION

This study provided an analysis of SVM and CNN for video forgery detection, offering insights into their effectiveness across multiple datasets. Unlike previous studies that focused on individual ML approaches, this research highlighted the advantages and limitations of both classifiers in detecting various types of forgeries. The experimental results demonstrated CNN's performance, achieving an accuracy up to 81.82%, significantly outperforming SVM, which struggled with larger datasets due to class imbalance and noise. CNN's exhibited a 15% improvement in localization precision and demonstrated robust adaptability in identifying both simple and complex forgery pattern, including intra-frame modifications and object-based alterations. The evaluation metrics further confirmed CNN's effectiveness in reducing FP and FNs, making it more reliable choice for forgery detection. These findings underscore the importance of DL-based approaches in video forgery detection, paving the way for more advanced, scalable, and reliable detection frameworks.

CONCLUSION

In order to demonstrate the growing demand for reliable identification techniques in digital security and forensic applications, this research compared SVM and CNN for video forgery identification. Traditional methods often fail to detect complex forgeries, especially in low-quality or compressed videos. The dataset underwent pre-processing steps, including resizing, grayscale conversion and normalization, to enhance uniformity and reduce noise. SVM utilized DOCFs to extract feature vectors, while CNN leveraged convolutional and pooling layers to capture intricate spatial-temporal forgery patterns such as edge inconsistencies and visual artifacts. Experimental results demonstrated CNN's superiority, achieving an accuracy of 73.33% on a dataset of 57 videos and 81.82% on a smaller set of 33 videos, significantly outperforming SVM's 46.67% accuracy. The findings emphasize CNN's capability to effectively capture motion and depth information for robust forgery detection, positioning it as a scalable and precise solution in evolving digital landscapes. Despite its reliance on larger data volumes, CNN proved to be a more reliable model across various evaluation metrics. Future research should explore enhancing detection accuracy in low-data scenarios by leveraging 3D-CNN architectures, which can effectively capture spatiotemporal features and improve forgery detection efficiency.

REFERENCES

[1]. Mahashreshty Vishweshwar, S. (2023). Implications of Deepfake Technology on Individual Privacy and Security.

[2]. Kingra, S., Aggarwal, N., & Kaur, N. (2023). Emergence of deepfakes and video tampering detection approaches: A survey. Multimedia Tools and Applications, 82(7), 10165-10209.

[3]. Huber, E., Pospisil, B., & Haidegger, W. (2021, May). Modus operandi in fake news. In 2021 IEEE Conference on Cognitive and Computational Aspects of Situation Management (CogSIMA) (pp. 127-132). IEEE.

[4]. Mubarak, R., Alsboui, T., Alshaikh, O., Inuwa-Dutse, I., Khan, S., & Parkinson, S. (2023). A survey on the detection and impacts of deepfakes in visual, audio, and textual formats. Ieee Access, 11, 144497-144529.

[5]. Ferreira, S., Antunes, M., & Correia, M. E. (2021). Exposing manipulated photos and videos in digital forensics analysis. Journal of Imaging, 7(7), 102.

[6]. Mohiuddin, S., Malakar, S., Kumar, M., & Sarkar, R. (2023). A comprehensive survey on state-of-the-art video forgery detection techniques. Multimedia Tools and Applications, 82(22), 33499-33539.

[7]. Sharma, S., & Dhavale, S. V. (2016, January). A review of passive forensic techniques for detection of copy-move attacks on digital videos. In 2016 3rd International conference on advanced computing and communication systems (ICACCS) (Vol. 1, pp. 1-6). IEEE.

[8]. Bilal, M., Habib, H. A., Mehmood, Z., Yousaf, R. M., Saba, T., & Rehman, A. (2021). A robust technique for copy-move forgery detection from small and extremely smooth tampered regions based on the DHE-SURF features and mDBSCAN clustering. Australian Journal of Forensic Sciences, 53(4), 459-482.

[9]. Hsu, C. C., Zhuang, Y. X., & Lee, C. Y. (2020). Deep fake image detection based on pairwise learning. Applied Sciences, 10(1), 370.
[10]. Farouk, M. A., & Fahmi, B. M. (2024). Deepfakes and media integrity: Navigating the new reality of synthetic content. Journal of Media and Interdisciplinary Studies, 3(9).
[11]. Alanazi, S., Asif, S., & Moulitsas, I. (2024). Examining the societal impact and legislative requirements of deepfake technology: a comprehensive study.
[12]. Hashmi, M. F., Ashish, B. K. K., Keskar, A. G., Bokde, N. D., Yoon, J. H., & Geem, Z. W. (2020). An exploratory analysis on visual counterfeits using conv-lstm hybrid architecture. IEEE Access, 8, 101293-101308.
[13]. Dang, L. M., Hassan, S. I., Im, S., & Moon, H. (2019). Face image manipulation detection based on a convolutional neural network. Expert Systems with Applications, 129, 156-168.
[14]. Sabir, E., Cheng, J., Jaiswal, A., AbdAlmageed, W., Masi, I., & Natarajan, P. (2019). Recurrent convolutional strategies for face manipulation detection in videos. Interfaces (GUI), 3(1), 80-87.
[15]. Tokas, B., Jakkinapalli, V. R., & Singla, N. (2023). Video Forgery Detection and Localization with Deep Learning Using W-NET Architecture. In Computational Intelligence: Select Proceedings of InCITe 2022 (pp. 31-38). Singapore: Springer Nature Singapore.
[16]. Rafique, R., Gantassi, R., Amin, R., Frnda, J., Mustapha, A., & Alshehri, A. H. (2023). Deep fake detection and classification using error-level analysis and deep learning. Scientific Reports, 13(1), 7422.
[17]. Raj, M., & Bakas, J. (2023, January). Detection of Object-Based Forgery in Surveillance Videos Utilizing Motion Residual and Deep Learning. In International Conference on Distributed Computing and Intelligent Technology (pp. 141-148). Cham: Springer Nature Switzerland.
[18]. Harika Palivela, L., Bala Gayathri, D., & Shanmuga Priya, R. (2023, February). Video Tampering Detection in Real Time. In International Conference on Intelligent Sustainable Systems (pp. 351-364). Singapore: Springer Nature Singapore.
[19]. Sohaib, M., & Tehseen, S. (2023). Forgery detection of low quality deepfake videos. Neural Network World, 33(2), 85.
[20]. Ding, X., Raziei, Z., Larson, E. C., Olinick, E. V., Krueger, P., & Hahsler, M. (2020). Swapped face detection using deep learning and subjective assessment. EURASIP Journal on Information Security, 2020, 1-12.