ISSN 2085-4552

Detection of AI-Generated Videos Using Local Binary Pattern and Support Vector Machine

Ergy David Lundy Tumanggor[1], Harlen Gilbert Simanullang[2], Arina Prima Silalahi[3]
Department of Informatics, Universitas Methodist Indonesia
Jl. Hang Tuah No.8, Madras Hulu, Kec. Medan Polonia, Kota Medan, Sumatera Utara 20151
ergydavid9145@ggmail.com[1], harlen.gilbert@gmail.com[2], primaarinasilalahi@gmail.com[3]

Accepted 29 May 2026
Approved 28 June 2026

***Abstract*— The rapid advancement of generative AI enables highly realistic fake videos, escalating financial and cybersecurity risks. As conventional verification methods become inadequate against evolving manipulations, this research proposes a robust, lightweight detection approach. The specific novelty of this study lies in repurposing a classic spatial texture extractor—Local Binary Pattern (LBP)—into a dynamic forensic tool by statistically aggregating frame-level micro-textures across the temporal axis to form a continuous 144-dimensional feature vector. This completely bypasses the massive computational overhead of deep learning. Utilizing the SDFVD2.0 dataset of 927 videos , frames undergo rigorous preprocessing, including face cropping and noise reduction, before LBP extraction. A linear Support Vector Machine (SVM) then classifies these vectors. The LBP-SVM model achieved an accuracy of 81.18% on the testing split and demonstrated swift processing (113.82 seconds for 40 unseen videos) during generalization testing. Although margins of error exist with highly complex artifacts , this spatial-temporal combination provides an efficient, interpretable, and computationally economical baseline for digital forensics.**

***Index Terms*— AI-Generated Video; Local Binary Pattern; Support Vector Machine.**

I. INTRODUCTION

The rapid advancement of generative artificial intelligence (AI) has enabled the creation of highly realistic fake videos that spread widely across social media and digital platforms. Consequently, it is increasingly difficult for the public to distinguish between genuine and fake videos, raising severe risks of financial fraud, disinformation, reputation attacks, privacy violations, and broader cybersecurity threats. Conventional inspection methods, such as manual visual verification or standard metadata analysis, are no longer adequate because digital manipulation techniques continue to evolve rapidly, seamlessly altering facial expressions, lip-syncing, and identities [1], [2], [3].

In the realm of digital forensics, numerous previous studies have tackled deepfake detection utilizing deep learning approaches based on convolutional neural networks (CNNs). For instance, research on deepfake detection using CNN architectures reported high performance, but these methods consistently demand massive datasets [3], [4], [5], [6], [7] , significant computational resources, and intensive transfer learning processes [8], [9], [10], [11], [12]. Conversely, in the broader field of machine learning, classic supervised learning methods like the Support Vector Machine (SVM) have proven highly reliable in predicting data classes [13] and are widely applied to various classification problems, including image and binary classifications [14]. Furthermore, classic image processing methods like Local Binary Pattern (LBP) have been successfully utilized to extract texture features for static object classification, such as dynamic video manipulation detection, material corrosion analysis, and cultural heritage pattern classification [15], [16], [17], [18], [19], [20].

Despite these advancements, there is a noticeable gap in the literature regarding the application of lightweight, classic machine learning techniques for dynamic video manipulation detection. While deep learning approaches like Convolutional Neural Networks (CNNs) offer robust automated feature extraction for digital forensics, they often demand massive datasets, significant computational resources, and intensive transfer learning processes [11], [12], [21]. These architectures typically require tens of millions of trainable parameters and high-end Graphical Processing Units (GPUs) for inference, often taking days to train and struggling to run in real-time on standard hardware. Conversely, this study introduces a distinct novelty by repositioning the classic spatial-temporal texture extraction of LBP combined with SVM as a highly efficient, interpretative baseline for dynamic AI-generated video detection [22], [23]. While LBP is a traditional image processing technique, its novelty here lies in its application to dynamic video

forensics. By statistically aggregating frame-level micro-textures—specifically the mean and standard deviation—across the temporal axis, this methodology transforms a static texture analyzer into a dynamic, 144-dimensional spatial-temporal feature extractor. This provides a highly efficient, interpretable baseline that solves the computational bottlenecks of CNNs, making it specifically designed for rapid verification on standard hardware devices with limited computing power.

By focusing on extracting the micro-texture of the facial area at the frame level using the SDFVD2.0 dataset, this study aims to close the existing gap by providing a simpler, classic texture feature approach. The proposed LBP-SVM model offers a lightweight, fast, and efficient baseline verification tool for digital forensics that can be easily implemented on devices with limited computing power, without the heavy burden of deep learning architectures.

A. System Architecture and Pipeline

This research adopts a chronological procedure starting from data collection, preprocessing, feature extraction, to SVM implementation. To provide a clear picture of the system architecture, the research framework is structured as a continuous workflow which can be seen in Fig. 1.

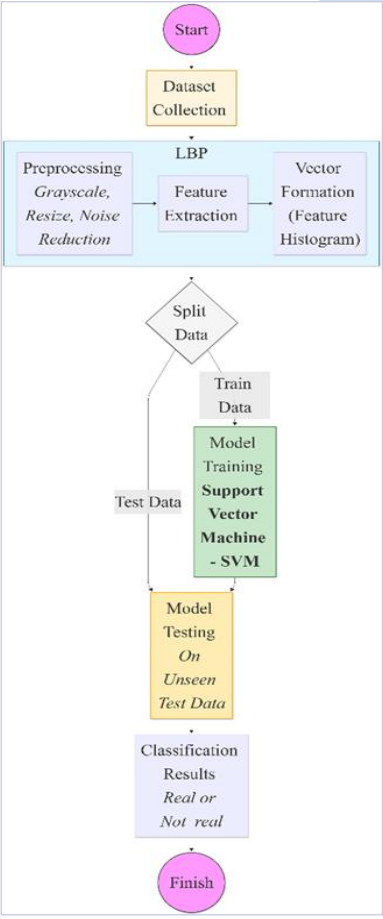

Fig. 1. Research Framework pipeline.

To formalize the research procedure, the systematic workflow of the proposed framework is detailed in Algorithm 1.

To ensure reproducibility and scientific validity, the sequential logic of the proposed LBP-SVM framework is mapped out visually. Instead of utilizing traditional algorithmic pseudocode, the systematic procedure of this research is comprehensively illustrated in Fig. 1.

B. Mathematical Formulation of Local Binary Pattern (LBP)

The feature extraction utilizes the Local Binary Pattern (LBP) method to capture local texture patterns. LBP is a gray-scale invariant texture measure that works by thresholding the neighborhood of each pixel and considers the result as a binary number. The LBP value is calculated by comparing the center pixel with its surrounding neighbors.

The mathematical calculation is defined in Equation (1):

$$LBP_{P,R} = \sum_{p=0}^{P-1} S(g_p - g_c)2^p \quad (1)$$

Where P is the number of neighbor points, and R is the radius of the neighborhood. The coordinates (x_c, y_c) represent the center pixel with a grayscale intensity of g_c, while g_p represents the intensity of the neighbor pixel. The function S(x) is a thresholding step function defined in Equation (2):

$$S(x) = 1, if\ x \geq 0\ ;\ S(x) = 0, if\ x < 0 \quad (2)$$

This system applies 16 LBP points (P=16) and a radius of 2 pixels (R=2), explicitly utilizing "uniform patterns." A binary pattern is considered uniform if it contains at most two bitwise transitions (from 0 to 1 or 1 to 0) in the circular representation of the pattern. The uniformity is calculated using Equation (3):

$$U(b) = \sum_{p=0}^{P-1} | b_p - b_{(p+1)}\ mod\ p | \quad (3)$$

Patterns where U(b) ≤ 2 are categorized as uniform, representing fundamental micro-textures like edges, corners, and flat regions, which are highly descriptive of facial manipulation artifacts.

C. Mathematical Formulation of Support Vector Machine (SVM)

As a supervised learning algorithm, the Support Vector Machine (SVM) works by finding the optimal decision boundary (hyperplane) to separate data into different classes with the maximum possible margin. In the broader scope of machine learning, choosing the right classification algorithm is crucial for handling complex features [14].

The linear SVM classification is based on finding a hyperplane that maximizes the margin between the two classes (real and not_real). The equation of the optimal hyperplane is defined in Equation (4):

$$w\ .\ x + b = 0 \quad (4)$$

Where w is the weight vector, x is the input feature vector (the 144-dimensional LBP features), and b is the bias.

To classify a new video data point, the model evaluates the features using the following decision function shown in Equation (5):

$$f(x) = sign(w \,.\, x + b) \quad (5)$$

If the output of f(x) is positive (+1), the data is classified into the first class (e.g., real). If the output is negative (-1), it is classified into the second class (e.g., not_real). SVM is highly suitable for this research due to its robustness in processing data within high-dimensional feature spaces without the heavy computational burden of deep learning architectures.

II. Result and Discussions

A. Preprocessing

Before extracting features, each video frame underwent a rigorous preprocessing pipeline (illustrated in Fig. 2) to ensure dimensional and format uniformity. This standard procedure is essential to enhance the precision of image-based classification tasks. The preprocessing stages included converting the original RGB frames (Fig. 2a) to grayscale (Fig. 2b), resizing them to a uniform 256x256 pixel resolution (Fig. 2c), applying a 3x3 Gaussian Blur to mitigate extreme compression artifacts (Fig. 2d), and finally, executing a face cropping operation (Fig. 2e) to isolate the Region of Interest (ROI).

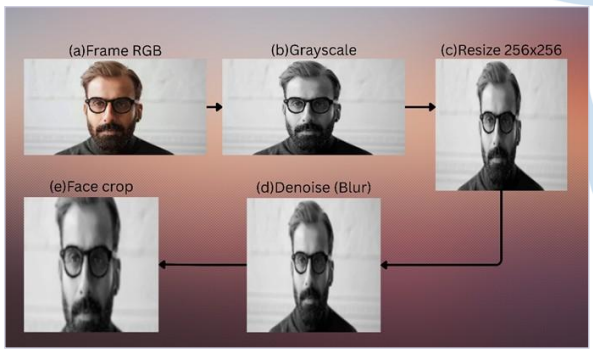

Fig. 2. Preprocessing stages on a single frame: (a) Original RGB, (b) Grayscale, (c) Resize 256x256, (d) Gaussian Blur Denoising, (e) Face Cropping.

This research adopts a chronological procedure starting from data collection, preprocessing, feature extraction, to SVM implementation. The process begins with video data collection from the SDFVD2.0 dataset, comprising 927 videos (456 real and 471 not_real). Because videos consist of multiple frames with high redundancy, a frame sampling mechanism is employed. These extracted frames then enter a rigid preprocessing pipeline to standardize the image format. To isolate the Region of Interest (ROI), an automated face detection algorithm using OpenCV is utilized [25]. In instances of face detection failure—often caused by extreme head angles, occlusions, or rapid motion—the system is programmed with a fallback mechanism to extract the center region of the frame (center crop) to ensure continuous feature extraction without injecting background noise. The isolated frames are then converted to a single-channel grayscale matrix, as the LBP operator only evaluates single-pixel intensities. Next, the frames are resized to a uniform 256x256 pixels. This specific resolution was selected to standardize the input matrix while retaining sufficient facial micro-textures without causing excessive memory consumption. Furthermore, a 3x3 Gaussian blur is applied. While blurring inherently reduces high-frequency data, a minimal 3x3 kernel was deliberately chosen as a necessary trade-off to smooth out benign, standard video compression artifacts that could mislead the classifier, without destroying the larger structural inconsistencies characteristic of AI manipulation.

B. Feature Extraction

Once the isolated 256x256 grayscale facial image is prepared, the LBP operator evaluates the micro-texture. The LBP parameters were specifically set to 16 neighbor points (P=16) and a radius of 2 pixels (R=2). Expanding the radius to R=2, utilizing precise bilinear interpolation to evaluate the exact intensities of diagonal neighbors, allows the algorithm to capture slightly larger, overlapping micro-textures. This spatial reach is critical for detecting the subtle, sub-pixel blending edges and warping artifacts typically left behind by generative AI algorithms at the boundaries of the face. To capture the dynamic changes across the video timeline, the Mean (μ) and Standard Deviation (σ) of these frame-level features are computed and concatenated into a final 144-dimensional continuous feature vector per video. The exact configuration is detailed in Table 1.

TABLE I. Feature Extraction Configuration

Parameter	Value
Image Size	256 x 256 pixels
LBP Points (P)	16
LBP Radius (R)	2
LBP Method	Uniform pattern
Bins per Grid	18
Grid Size	2 x 2 = 4 grids
Final Features	144 (72 Mean + 72 Std Dev)

To capture the video's temporal dynamics, both the mean (μ) and standard deviation (σ) of these features are calculated across all frames, generating a final continuous feature vector of 144 dimensions. Utilizing statistical features like mean and standard deviation from image matrices has been highly effective in various texture classification approaches [26].

C. Model Training and Evaluation

For classification, the dataset of 144-dimensional continuous features is normalized using Z-score standardization. A Linear Support Vector Machine (SVM) is then trained on 80% of the data, utilizing a penalty parameter of C=10.0 and balanced class weights to define the optimal decision boundary [13]. A Linear SVM was specifically selected over non-linear kernels due to the high-dimensional nature of the extracted data. In 144-dimensional spaces, data points are often already linearly separable; applying a complex, non-linear kernel across a relatively compact dataset poses a severe risk of overfitting to the training distribution. Therefore, the linear kernel prevents overfitting while maintaining mathematical interpretability, allowing for transparent observation of the feature weights dictating the decision boundary.

During training, the SVM resolved the Dual Lagrange formulation to identify the optimal hyperplane. Out of the 741 training samples, the algorithm identified precisely 372 critical data points as Support Vectors (data points where the Lagrange multiplier $\alpha > 0$). These vectors dictate the exact position of the decision boundary. By mathematically verifying the Karush-Kuhn-Tucker (KKT) conditions, it was observed that bounded support vectors precisely hit the penalty limits (e.g., $\alpha = 9.623$ for manipulated samples), while free support vectors lay exactly on the margin boundaries. The accumulation of the weight vector components resulted in an optimal geometric margin width of Margin = 2 / ||w|| = 0.1151. To better understand the feature distribution and the classification boundary, the 144-dimensional feature space was projected into a 2D visualization using dimensionality reduction, as shown in Fig. 3.

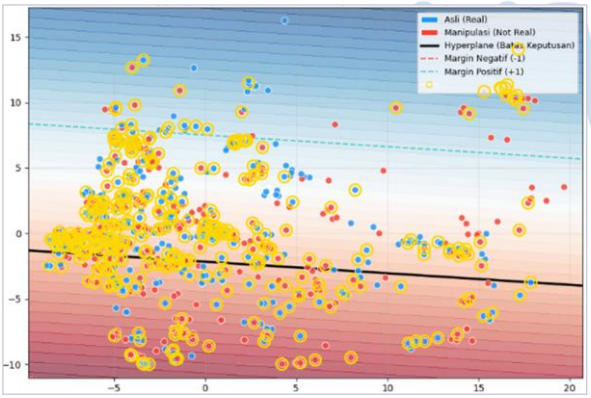

Fig. 3. SVM Hyperplane, Margin boundaries, and Support Vectors projection.

In Fig. 3, the solid black line represents the optimal linear decision boundary separating the real class from the AI-generated class. The dashed lines illustrate the margin boundaries for each class. The data points highlighted with yellow circles are the Support Vectors, which are the critical samples resting exactly on or violating the margin bounds. The mathematical accumulation of these support vectors resulted in a calculated optimal margin width of 0.1151, maximizing the distance between the two classes to lower the generalization error. The model evaluation on the testing data yielded an overall accuracy of 81.18%. The detailed performance metrics are presented in Table 2.

TABLE II. CLASSIFICATION REPORT

Metric	Precision	Recall	F1-Score	Support
real	0.8283	0.8200	0.8241	100
not_real	0.7931	0.8023	0.7977	86
Accuracy			0.8118	186

To provide deeper insight into the model's classification behavior and to detail the exact distribution of true and false predictions across the testing split, a confusion matrix was generated as presented in Fig 4.

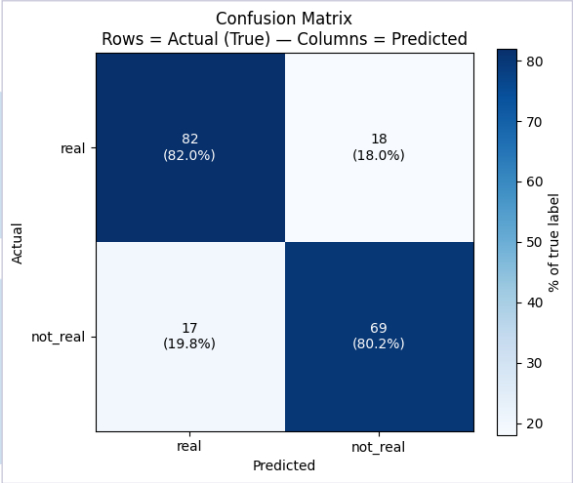

Fig. 4. SVM Hyperplane, Margin boundaries, and Support Vectors projection.

D. Testing on Unseen Data

To rigorously validate the model's generalization capabilities in real-world scenarios, it was tested on 40 completely new video datasets not included in the training or validation splits. The final decision function for the linear kernel is formulated in equation (6).

$$f(x_{new}) = sign(\sum_{i=1}^{144}(w_i x_{i,new}) + b \qquad (6)$$

Where w_i are the learned weights, $x_{i,new}$ are the scaled features of the new video, and b is the learned bias (calculated as 0.2234). If the output is positive, the video is classified as real; if negative, it is classified as not_real. The results of this unseen data testing are summarized in Table 3.

TABLE III. PREDICTION RESULTS SUMMARY

No	Video	Prediksi	Conf.	P(real)	P(fake)	Frame	Waktu
1	video_1_not_real.mp4	not_real	62.54%	37.46%	62.54%	362	7.63s
2	video_1_real.mp4	not_real	65.58%	34.42%	65.58%	81	1.74s

3	video_2_not_real.mp4	real	50.74%	50.74%	49.26%	307	5.85s
4	video_2_real.mp4	real	52.62%	52.62%	47.38%	81	1.73s
5	video_3_not_real.mp4	not_real	80.91%	19.09%	80.91%	82	1.77s
6	video_3_real.mp4	not_real	56.66%	43.34%	56.66%	321	10.39s
7	video_4_not_real.mp4	not_real	62.41%	37.59%	62.41%	82	2.07s
8	video_4_real.mp4	real	67.55%	67.55%	32.45%	60	1.64s
9	video_5_not_real.mp4	not_real	65.58%	34.42%	65.58%	120	3.02s
10	video_5_real.mp4	real	76.11%	76.11%	23.89%	244	13.17s
11	video_6_not_real.mp4	not_real	66.73%	33.27%	66.73%	120	2.58s
12	video_6_real.mp4	real	51.78%	51.78%	48.22%	81	1.74s
13	video_7_not_real.mp4	not_real	59.59%	40.41%	59.59%	102	2.02s
14	video_7_real.mp4	real	53.64%	53.64%	46.36%	120	2.78s
15	video_8_not_real.mp4	not_real	56.56%	43.44%	56.56%	102	2.28s
16	video_8_real.mp4	real	65.68%	65.68%	34.32%	120	2.85s
17	video_9_not_real.mp4	not_real	60.63%	39.37%	60.63%	127	3.00s
18	video_9_real.mp4	real	54.32%	54.32%	45.68%	120	2.89s
19	video_10_not_real.mp4	not_real	55.93%	44.07%	55.93%	127	2.50s
20	video_10_real.mp4	not_real	54.55%	45.45%	54.55%	94	1.98s
21	video_11_not_real.mp4	real	51.18%	48.82%	51.18%	171	3.26s
22	video_11_real.mp4	not_real	55.67%	44.33%	55.67%	94	1.98s
23	video_12_not_real.mp4	not_real	55.57%	44.43%	55.57%	171	3.24s
24	video_12_real.mp4	not_real	54.12%	45.88%	54.12%	94	1.95s
25	video_13_not_real.mp4	real	50.77%	50.77%	49.23%	118	2.52s
26	video_13_real.mp4	real	54.00%	54.00%	46.00%	117	2.43s
27	video_14_not_real.mp4	not_real	53.14%	46.86%	53.14%	118	2.53s
28	video_14_real.mp4	real	50.97%	49.03%	50.97%	117	2.43s
29	video_15_not_real.mp4	not_real	58.11%	41.89%	58.11%	100	2.11s
30	video_15_real.mp4	real	53.65%	53.65%	46.35%	117	2.29s
31	video_16_not_real.mp4	not_real	58.11%	41.89%	58.11%	100	2.12s
32	video_16_real.mp4	real	53.65%	53.65%	46.35%	99	1.87s
33	video_17_not_real.mp4	not_real	56.95%	43.05%	56.95%	93	1.96s
34	video_17_real.mp4	real	58.17%	58.17%	41.83%	94	1.93s
35	video_18_not_real.mp4	not_real	52.29%	47.71%	52.29%	93	2.03s
36	video_18_real.mp4	real	63.64%	63.64%	36.36%	94	1.95s
37	video_19_not_real.mp4	real	58.78%	58.78%	41.22%	97	2.12s
38	video_19_real.mp4	real	60.54%	60.54%	39.46%	94	1.95s
39	video_20_not_real.mp4	real	51.53%	48.47%	51.53%	97	2.15s
40	video_20_real.mp4	real	55.82%	55.82%	44.18%	115	2.35s

The system successfully and accurately predicted 30 out of 40 videos, achieving a generalization accuracy of 75%. The model demonstrated an average confidence rate of 58.43% with a total processing time of 113.82 seconds for all 40 videos combined. This indicates that when the model encounters borderline data with highly complex textures that deviate significantly from the training distribution, it becomes mathematically uncertain.

Despite these margin errors, the findings substantiate that the spatial-temporal combination of LBP and SVM provides a highly efficient, transparent, and lightweight baseline for digital forensics. This approach operates effectively and rapidly on devices with limited computing power, offering a practical alternative to computationally heavy CNN architectures.

E. Model Evaluation and Error Analysis

While the LBP-SVM framework demonstrated high processing efficiency, an in-depth evaluation reveals key vulnerability margins in the classification boundaries. The misclassification of **10 videos** (such as *video_2_not_real*, *video_11_real*, and *video_20_not_real*) during generalization testing highlights the specific limitations of spatial-temporal LBP when confronting advanced generative manipulation. Noticeably, the confidence scores for these misclassified instances tend to hover closely around the borderline margin (50% to 56%), demonstrating that when the model encounters highly complex textures that deviate significantly from the training distribution, it becomes mathematically uncertain.

Error analysis indicates that these failures stem from highly complex artifacts that deviate from the fundamental micro-textures (edges, corners, and flat regions) captured by the uniform pattern of $U(b) \leq 2$. Specifically, erratic environmental lighting and subtle sub-pixel blending applied by newer AI generation methods create non-uniform local binary patterns that push the feature vectors directly onto the decision margin, creating mathematical uncertainty for the linear hyperplane. This demonstrates that while the SVM effectively separates standard manipulation artifacts, its linear separation struggles with highly non-linear, multi-illuminated synthetic textures.

The implementation of Local Binary Pattern (LBP) and Support Vector Machine (SVM) successfully establishes a scientifically rigorous, mathematically sound, and highly efficient baseline for detecting AI-generated videos. The proposed model, utilizing a 144-dimensional temporal-aggregated feature vector,

achieved a solid accuracy of 81.18% on the testing split. Furthermore, it correctly classified 30 out of 40 completely unseen videos during generalization testing, yielding an average confidence rate of 58.43% with an incredibly rapid total processing time of 113.82 seconds. These empirical results confirm that the LBP-SVM framework provides a lightweight, fast, and computationally economical verification tool, making it highly practical for on-the-fly digital forensic applications operating on standard hardware.

Despite its computational efficiency and transparency, this classic spatial-temporal texture extraction approach encounters certain limitations. The model faces challenges—exhibiting lower confidence and margin errors—when classifying videos containing highly complex textures, erratic lighting, or subtle manipulation artifacts that deviate significantly from the baseline training dataset. To address these limitations and advance future research, it is highly recommended to explore the integration or comparative analysis of this conventional LBP-SVM method with advanced Deep Learning architectures [27] to enhance granular feature extraction capabilities. Furthermore, expanding the dataset to include synthetic media from newer generative AI models (such as diffusion models), and evaluating the system under more diverse real-world conditions (varying lighting environments, differing resolutions, and diverse visual angles) will be crucial to strengthening the overall robustness and reliability of the detection system. Finally, deploying this trained model into a user-friendly application interface (UI/UX) is recommended to facilitate accessibility for end-users in real-world forensic investigations.

REFERENCES

[1] A. A. Khan, A. A. Laghari, S. A. Inam, S. Ullah, M. Shahzad, and D. Syed, "A survey on multimedia-enabled deepfake detection: state-of-the-art tools and techniques, emerging trends, current challenges & limitations, and future directions," Dec. 01, 2025, *Springer Science and Business Media B.V.* doi: 10.1007/s10791-025-09550-0.

[2] R. Tolosana, R. Vera-Rodriguez, J. Fierrez, A. Morales, and J. Ortega-Garcia, "DeepFakes and Beyond: A Survey of Face Manipulation and Fake Detection," Jun. 2020, [Online]. Available: http://arxiv.org/abs/2001.00179

[3] Y. Li and S. Lyu, "Exposing DeepFake Videos By Detecting Face Warping Artifacts," *arXiv*, May 2019, [Online]. Available: http://arxiv.org/abs/1811.00656

[4] A. Rössler, D. Cozzolino, L. Verdoliva, C. Riess, J. Thies, and M. Nießner, "FaceForensics++: Learning to Detect Manipulated Facial Images," *arXiv*, Aug. 2019, [Online]. Available: http://arxiv.org/abs/1901.08971

[5] D. Afchar, V. Nozick, J. Yamagishi, and I. Echizen, "MesoNet: a Compact Facial Video Forgery Detection Network," *arXiv*, Sep. 2018, doi: 10.1109/WIFS.2018.8630761.

[6] I. Masi, A. Killekar, R. M. Mascarenhas, S. P. Gurudatt, and W. AbdAlmageed, "Two-branch Recurrent Network for Isolating Deepfakes in Videos," *arXiv*, Sep. 2020, [Online]. Available: http://arxiv.org/abs/2008.03412

[7] S. M. Yasir and H. Kim, "Lightweight Deepfake Detection Based on Multi-Feature Fusion," *Appl. Sci.*, vol. 15, no. 4, Feb. 2025, doi: 10.3390/app15041954.

[8] J. C. C, O. J. A, N. M.D, S. Ahmad, F. O. O, and U. A. A, "Development Of Artificial Intelligence-Based Model for Forensic Analysis of Cross-Platform Deepfakes," *Int. J. Res. Sci. Innov.*, 2025, doi: 10.51244/IJRSI.

[9] J. Mallet, L. Pryor, D. R. Dave, and D. M. Vanamala, "Deepfake Detection Analyzing Hybrid Dataset Utilizing CNN and SVM," 2023.

[10] Y. Jugade, Z. Syed, C. Dcosta, A. Panicker, and D. Vora, "Deepfake detection via spatial–temporal deep networks: leveraging CNNs and LSTMs for enhanced accuracy," *Discov. Appl. Sci.*, Jan. 2026, doi: 10.1007/s42452-025-08014-w.

[11] S. A. Falcón-López, L. Tobarra, A. Robles-Gómez, and R. Pastor-Vargas, "Forensic Analysis of Manipulated Images and Videos," *MDPI*, vol. 15, no. 23, Dec. 2025, doi: 10.3390/app152312664.

[12] A. Raza *et al.*, "A Comprehensive Review of Deepfake Detection Techniques: From Traditional Machine Learning to Advanced Deep Learning Architectures," *MDPI*, vol. 7, no. 2, Feb. 2026, doi: 10.3390/ai7020068.

[13] Andi Saenong and R. Rahmat, "Implementasi Algoritma SVM Untuk Sistem Deteksi dan Pengawasan Keamanan Kendaraan di Area Parkir Menggunakan Kamera," *Simkom*, vol. 9, no. 2, pp. 267–277, Jul. 2024, doi: 10.51717/simkom.v9i2.570.

[14] A. P. Silalahi and H. G. Simanullang, *DECISSION TREE DAN PENERAPANNYA Dengan Algoritma Iterative Dichotomizes versi 3 (ID3)*. Madza Media, 2025. [Online]. Available: www.madzamedia.co.id

[15] Y. Chen, N. Akhtar, N. A. H. Haldar, and A. Mian, "Deepfake Detection with Spatio-Temporal Consistency and Attention," *Univ. West. Aust.*, Feb. 2025, [Online]. Available: http://arxiv.org/abs/2502.08216

[16] J. Jiang, W. C. Yang, C. H. Chen, and T. Young, "A New Deepfake Detection Method with No-Reference Image Quality Assessment to Resist Image Degradation," *MDPI*, vol. 6, no. 10, Oct. 2025, doi: 10.3390/eng6100274.

[17] C. Internò, R. Geirhos, M. Olhofer, S. Liu, B. Hammer, and D. Klindt, "AI-Generated Video Detection via Perceptual Straightening," *arXiv*, Feb. 2026, [Online]. Available: http://arxiv.org/abs/2507.00583

[18] F. Bimantoro, R. Nisful, L. Hidhayah, and D. Swanjaya, "Komparasi Algoritma MLP+LBP dan CNN Sebagai solusi Inovatif Untuk Deteksi Dini Korosi," *INOTEK*, vol. 8, pp. 2549–7952, 2024.

[19] S. K. Dirjen, P. Riset, D. Pengembangan, R. Dikti, P. N. Andono, and E. H. Rachmawanto, "Evaluasi Ekstraksi Fitur GLCM dan LBP Menggunakan Multikernel SVM untuk Klasifikasi Batik," *J. Resti*, no. 3, 2021, doi: 10.29207/resti.v5i1.265.

[20] N. S. Fatimah and S. Agustin, "Klasifikasi Citra Batik Menggunakan Local Binary Pattern (LBP) dan Support Vector Machine (SVM)," *J. Algoritm.*, 2025.

[21] M. D. Rahadian, A. Aziz, and D. W. Prabowo, "Performance Analysis of the LBP-SVM Model in Deepfake Image Detection: A Case Study on the FaceForensics++ Dataset and External Validation on Indonesian Politicians' Faces."

[22] S. M. Coloma *et al.*, "Deepfake and Forged Image Detection through Enhanced Local Binary Pattern Histogram (eLBPH) Features and SVM Classification for Digital Image Forensics," 2025.

[23] K. Azmi, S. Defit, and Sumijan, "Implementasi Convolutional Neural Network (CNN) Untuk Klasifikasi Batik Tanah Liat Sumatera Barat," *J. Unitek*, vol. 16, no. 1, p. 2023, 2023.

[24] A. I. Nurulrachman, R. Cahya Wihandika, and P. P. Adikara, "Ekstraksi Ciri pada Klasifikasi Citra Batik menggunakan Metode Gray Level Co-Occurrence Matrix, Local Binary Pattern, dan HSV Color Moment," 2023. [Online]. Available: http://j-ptiik.ub.ac.id

[25] R. H. P. Sejati and R. Mardhiyyah, "DETEKSI WAJAH BERBASIS FACIAL LANDMARK MENGGUNAKAN OPENCV DAN DLIB," *J. Teknol. Inf.*, vol. 5, no. 2, 2021.

[26] B. Zi, M. Chang, J. Chen, X. Ma, and Y. G. Jiang, "WildDeepfake: A Challenging Real-World Dataset for Deepfake Detection," in *MM 2020 - Proceedings of the 28th ACM International Conference on Multimedia*, Association for Computing Machinery, Inc, Oct. 2020, pp. 2382–2390. doi: 10.1145/3394171.3413769.

[27] L. Guarnera, O. Giudice, and S. Battiato, "Fighting deepfake by exposing the convolutional traces on images," *IEEE Access*, vol. 8, pp. 165085–165098, 2020, doi: 10.1109/ACCESS.2020.3023037.