A Study on Image and Video Forgery Detection Techniques

Thesis submitted by

Sk Mohiuddin

Doctor of Philosophy (Engineering)

Department of Computer Science and Engineering
Faculty Council of Engineering & Technology
Jadavpur University
Kolkata-700032, India
2025

JADAVPUR UNIVERSITY
KOLKATA 700032, INDIA

Index No: 120/22/E

1. Title of the Thesis:

   A Study on Image and Video Forgery Detection Techniques

2. Name, Designation & Institute of the supervisor:

   Prof. Ram Sarkar
   Professor,
   Department of Computer Science and Engineering,
   Jadavpur University, Kolkata-700032

   Dr. Samir Malakar
   Assistant Professor,
   Department of Computer Science,
   Asutosh College, Kolkata-700026

3. List of Publications:

   a) Journal

   i) Shreyan Ganguly, **Sk Mohiuddin**, Samir Malakar, Erik Cuevas, Ram Sarkar. "Visual attention-based deepfake video forgery detection", ***Pattern Analysis and Applications (2022) (IF:3.7)***.

   ii) Shreyan Ganguly, Aditya Ganguly, **Sk Mohiuddin**, Samir Malakar, Ram Sarkar. "ViXNet: Vision Transformer with Xception Network for deepfakes based video and image forgery detection", ***Expert Systems with Applications (2022) (IF: 7.5)***.

   iii) **Sk Mohiuddin**, Samir Malakar, Ram Sarkar. "A comprehensive survey on state-of-the-art video forgery detection techniques", ***Multimedia Tools and Applications (2023) (IF: 3.0)***.

   iv) **Sk Mohiuddin**, Khalid Hassan Sheikh, Samir Malakar, Juan D Velásquez, Ram Sarkar. "A hierarchical feature selection strategy for deepfake video detection", ***Neural Computing and Applications (2023) (IF:4.8)***.

v) **Sk Mohiuddin**, Samir Malakar, Ram Sarkar. "An ensemble approach to detect copy-move forgery in videos", ***Multimedia Tools and Applications (2023) (IF:3.0)***.

vi) Gourab Naskar, **Sk Mohiuddin**, Samir Malakar, Ram Sarkar. "Deepfake detection using deep feature stacking and meta-learning", ***Heliyon (2024) (IF: 3.4)***.

vii) Ayush Roy, **Sk Mohiuddin**, Ram Sarkar. "A Similarity-based Positional Attention aided Deep Learning Model for Copy-Move Forgery Detection", ***IEEE Transactions on Artificial Intelligence (2024).***

viii) **Sk Mohiuddin**, Ayush Roy, Saptarshi Pani, Samir Malakar, Ram Sarkar. "A feature selection-aided deep learning based deepfake video detection method", ***Multimedia Tools and Applications (2025): 1-24. (IF:3.0)***.

b) **Conference**

i) **Sk Mohiuddin**, Shreyan Ganguly, Samir Malakar, Dmitrii Kaplun, Ram Sarkar. "A feature fusion based deep learning model for deepfake video detection", In ***International Conference on Mathematics and its Applications in new Computer Systems (MANCS 2021). Lecture Notes in Networks and Systems, vol 424. Springer, Cham.*** https://doi.org/10.1007/978-3-030-97020-8_18

ii) **Sk Mohiuddin**, Samir Malakar, Ram Sarkar. "Duplicate frame detection in forged videos using sequence matching", In ***International Conference on Computational Intelligence in Communications and Business Analytics (CICBA 2021). Communications in Computer and Information Science, vol 1406. Springer, Cham.*** https://doi.org/10.1007/978-3-030-75529-4_3

iii) Ayush Roy, **Sk Mohiuddin**, Maxim Minenko, Dmitrii Kaplun, and Ram Sarkar. "Deepspace: Navigating the frontier of deepfake identification using attention-driven xception and a task-specific subspace", In ***In Proceedings of the 20th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP 2025). Volume 2: VISAPP, pages 163-172.*** https://doi.org/10.5220/0013173700003912

4. List of Patents: None

5. List of Presentations in National/International/Conference/Workshops:

   i) **Sk Mohiuddin**, Shreyan Ganguly, Samir Malakar, Dmitrii Kaplun, Ram Sarkar. "A feature fusion based deep learning model for deepfake video detection", In ***International Conference on Mathematics and its Applications in new Computer Systems (MANCS 2021). Lecture Notes in Networks and Systems, vol 424. Springer, Cham.*** https://doi.org/10.1007/978-3-030-97020-8_18

   ii) **Sk Mohiuddin**, Samir Malakar, Ram Sarkar. "Duplicate frame detection in forged videos using sequence matching", In ***International Conference on Computational Intelligence in Communications and Business Analytics (CICBA 2021). Communications in Computer and Information Science, vol 1406. Springer, Cham.*** https://doi.org/10.1007/978-3-030-75529-4_3

Statement of Originality

I, **Sk Mohiuddin** registered on **18/05/2022** do hereby declare that this thesis entitled **"A Study on Image and Video Forgery Detection Techniques"** contains literature survey and original research work done by the undersigned candidate as part of doctoral studies.

All information in this thesis have been obtained and presented in accordance with existing academic rules and ethical conduct. I declare that, as required by these rules and conduct, I have fully cited and referred all materials and results that are not original to this work.

I also declare that I have checked this thesis as per the "Policy on Anti Plagiarism, Jadavpur University, 2019", and the level of similarity as checked by iThenticate software is 4%.

Signature of Candidate

Date:

Certified by Supervisors:
(Signature with date and seal)

1.

2.

PROFORMA-2

CERTIFICATE FROM THE SUPERVISOR

Index No: 120/22/E

This is certify that the thesis entitled **"A Study on Image and Video Forgery Detection Techniques"** submitted by Sk Mohiuddin, who got his name registered on 18/05/2022, for the award of Ph.D. (Engg.) degree of Jadavpur University is absolutely based upon his own work under the supervision of Dr. Ram Sarkar, Professor, Department of Computer Science and Engineering, Jadavpur University, Kolkata, India and Dr. Samir Malakar, Assistant Professor, Department of Computer Science, Asutosh College, Kolkata, India that neither his thesis nor any part of it has been submitted for any degree/diploma or any other academic award anywhere before.

1 .....

Dr. Ram Sarkar
Professor,
Department of Computer Sc. and Engineering,
Jadavpur University,
Kolkata-700032

2 .....

Dr. Samir Malakar
Assistant Professor,
Department of Computer Science,
Asutosh College,
Kolkata-700026

Dedicated to my Family

ACKNOWLEDGMENT

The successful completion of this research would not have been possible without the guidance, support, and encouragement of several individuals. I take this opportunity to express my deepest gratitude to all those who have contributed to this work, both directly and indirectly.

First and foremost, I would like to extend my sincere appreciation and deep respect to my supervisors, **Prof. Ram Sarkar**, Department of Computer Science and Engineering, Jadavpur University, Kolkata, India and **Dr. Samir Malakar**, Assistant Professor, Department of Computer Science, Asutosh College, Kolkata, India for their unwavering guidance, immense support, continuous encouragement, invaluable insights, and dedicated supervision throughout this journey. Their expertise and mentorship have been instrumental in shaping the direction of this research, and I am truly grateful for their patience and belief in my work.

I would also like to extend my sincere respect and gratitude to Prof. Nirmalya Chowdhury, Head of the Department of Computer Science and Engineering, Jadavpur University, as well as to Prof. Mahantapas Kundu, Prof. Debotosh Bhattacharjee, Prof. Subhadip Basu and Prof. Nandini Mukherjee from the same department for their continuous encouragement, invaluable support, and insightful suggestions throughout the course of this thesis. Additionally, I am grateful to the faculty and staff members of the Department of Computer Science and Engineering, Jadavpur University, for their assistance and support during the course of my research work.

I am deeply grateful to Dr. Neelotpal Chakraborty and Dr. Soumyendu Sekhar Bandyopadhyay for their encouragement and support in my research. I also extend my heartfelt thanks to my co-researchers Mr. Shreyan Ganguly, Mr. Aditya Ganguly, Mr. Khalid Hassan Sheikh, Mr. Gourab Naskar and Mr. Ayush Roy for their unwavering support during my research journey. Additionally, I sincerely appreciate the encouragement and solidarity of my fellow researchers, including Mr. Asfak Ali, Ms. Payel Pramanik, Ms. Nandita Gautam, Mr. Sujan Sarkar and Mr. Sourajit Maity, from the CMATER Laboratory, Jadavpur University, Kolkata, India.

I sincerely thank my parents for their unwavering support throughout my academic journey and my parents-in-law for their encouragement throughout the course of my research work. Their belief in me and sacrifices have been invaluable. I am truly fortunate to have their constant love and support. Their strength and kindness have been a source of inspiration throughout this journey.

I am also deeply thankful to my in-laws, my beloved wife Sabera, our wonderful daughter Zikra, and our dear son Abdur Rahman for their unconditional love and unwavering support. Their presence has been a constant source of strength and motivation throughout this challenging academic endeavour.

Finally, I offer my sincere gratitude to the Almighty for His countless blessings, which have guided me in making this journey a success.

Sk Mohiuddin

Index No: 120/22/E
Registration No: 1021904005 of 2022-2023
Department of Computer Science and Engineering,
Jadavpur University, Kolkata, India.

Contents

List of Figures xix

List of Tables xxv

List of Acronyms/Abbreviations xxix

1 Introduction 1

1.1 Importance of Digital Media Integrity . . . 3
1.2 Rise of Image and Video Forgeries . . . 4
1.3 Classification of Image Forgery Techniques . . . 6
1.4 Classification of Video Forgery Techniques . . . 8
1.5 Problem Definition: Challenges Posed by Various Types of Forgeries . . 11
1.6 Objectives of the Thesis . . . 13
1.7 Scope of the Thesis . . . 14
1.8 Organization of the Thesis . . . 17

2 Literature Review 19

2.1 Image Copy-Move Forgery Detection Techniques . . . 23
2.1.1 Keypoint-based Approach . . . 24
2.1.2 Block-based Approach . . . 25
2.1.3 Deep Learning-based Approach . . . 26
2.2 Deepfake Detection Techniques . . . 29
2.3 Inter-frame Video Forgery Detection Techniques . . . 32
2.4 Discussion . . . 34

3 Copy-Move Forgery Detection in Images 35

3.1 Methodology . . . 36
3.1.1 MultiResUNet . . . 37
3.1.2 SPAM . . . 38
3.1.3 Evaluation Metrics . . . 41
3.1.4 Loss Function . . . 41
3.1.5 Experimental Setup . . . 42

3.1.6 Dataset Description . . . 42
3.1.7 Ablation Study . . . 43
3.1.8 Comparison with Past Methods . . . 45
3.1.9 Error Analysis . . . 47
3.2 Discussion . . . 48

4 Deepfake Detection in Images/Videos 49
4.1 Methodology 1: ViXNet - A Deep Learning-based Model for Deepfake Detection . . . 50
4.1.1 Patch-wise Self-Attention Module . . . 51
4.1.2 Global Self-Attention Module . . . 52
4.1.3 Global Image Feature Extraction . . . 53
4.1.4 Classification Model . . . 54
4.2 Experimental Results and Discussion . . . 54
4.2.1 Dataset used for the Proposed Methods . . . 55
4.2.2 Data Preparation . . . 56
4.2.3 Experimental Setups . . . 57
4.2.4 Model Parameters . . . 58
4.2.5 Results using Intra-dataset Experimental Setup . . . 58
4.2.6 Results using Inter-dataset Experimental Setup . . . 59
4.2.7 Comparison with Past Methods . . . 62
4.2.8 Results on the DFDC Dataset . . . 65
4.2.9 Error Analysis . . . 66
4.3 Methodology 2: Hierarchical Feature Selection - based Deepfake Detection using Combined Deep and Handcrafted Features . . . 67
4.3.1 Feature Extraction . . . 68
4.3.2 Feature Selection . . . 69
4.3.3 Grey Wolf Optimizer . . . 69
4.3.4 Vortex Search Algorithm . . . 71
4.3.5 Proposed Hybrid FS Method . . . 72
4.3.6 Fitness Function . . . 73
4.3.7 Transfer Function . . . 74
4.3.8 Hierarchical Feature Selection . . . 75
4.4 Results and Discussion . . . 75
4.4.1 Experimental Setup . . . 75
4.4.2 Experimental Results . . . 76
4.4.3 Inter-Dataset Generalization Capability of Selected Features . . . 80

4.4.4 Inter-Dataset Generalization Capability of the Proposed HFS Method . . . 81
4.4.5 Comparison with Other FS Methods . . . 82
4.4.6 Comparison with Past Deepfake Detection Methods . . . 82
4.4.7 Performance on the DFDC Dataset . . . 85
4.5 Discussion . . . 85

5 Inter-frame Video Forgery Detection 87
5.1 Frame Duplication Detection Technique . . . 88
5.1.1 Measuring Similarity between Two Frames . . . 89
5.1.2 Detection of Shots in a Video . . . 90
5.1.3 Frame Duplication Detection . . . 90
5.2 Experiment Results and Analysis . . . 92
5.2.1 Dataset Design for Frame Duplication Attack Detection . . . 92
5.2.2 Evaluation Metrics . . . 93
5.2.3 Results and Discussion . . . 93
5.3 Frame Deletion Forgery Detection . . . 96
5.3.1 Preprocessing . . . 97
5.3.2 Detection of Frame Deletion Attack . . . 97
5.3.3 Localization of Deleted Frames . . . 99
5.4 Experimental Evaluation and Observations . . . 100
5.4.1 Dataset Description . . . 101
5.4.2 Evaluation Metrics . . . 102
5.4.3 Experimental Setup . . . 103
5.4.4 Performance of Frame Deletion Attack Detection . . . 103
5.4.5 Performance of Deleted Frames Localization . . . 104
5.4.6 Performance of the Proposed Method . . . 105
5.4.7 Performance on Post-processed Videos . . . 105
5.4.8 Analysis of Failure Cases . . . 106
5.5 Discussion . . . 108

6 Conclusion and Future Scope 111

References 119

List of Figures

1.1 (a) Real video showing an unaltered, authentic clip of former President Barack Obama. (b) Fake video generated by AI software that creates highly realistic deepfake footage of Obama using existing audio and video clips [9]. 2

1.2 Snapshots of news articles highlighting the societal impact and risks associated with manipulated media, reflecting growing public and academic concern. 4

1.3 Lincoln's face from the five-dollar bill photo was flipped and placed onto a slavery proponent's body from an 1852 engraving [20]. 7

1.4 "The Commissar Vanishes", David King's visual history of the falsification of images, explores how Stalin manipulated photography to erase all memory of his victims [21]. 7

1.5 (a) Real Image: The unaltered, authentic photograph, (b) Fake Image: A digitally manipulated version of the original [22]. 7

1.6 Classification of image forgery techniques. 8

1.7 Hierarchical classification of video forgery techniques. 9

1.8 General types of intra-frame video forgeries (a) original frame, (b) fake frame [31]. 9

1.9 Upscale video forgery process (top row): (a) original video frame, (b) forged video frame created by upscaling the original to conceal boundary areas (highlighted in red). 10

1.10 General types of inter-frame video forgeries. (a) Original frame sequence, (b) Frame insertion from another video, (c) Frame deletion, and (d) Frame duplication (video copy-move forgery). 11

1.11 Examples of real facial images alongside their corresponding Deepfake versions. 12

2.1 Classification of image/video forgery detection techniques. 20

3.1 Examples of copy-move forgery images are shown, where the first column displays the original images, the second column shows the forged images, and the third column presents the corresponding ground truth masks [204]. 36

3.2 The block diagram illustrates the proposed modified MultiResUNet architecture incorporating the SPAM attention module. MultiRes blocks (MRB) are employed to extract meaningful features from the input image. Residual connections (RB) between encoder and decoder features ensure efficient information flow across multiple network levels. The attention-enhanced bottleneck feature map undergoes upsampling (UP) and is subsequently multiplied with the decoder feature map to refine model performance. Finally, a convolutional layer with sigmoid activation (Conv) generates the predicted binary segmentation mask. . 37

3.3 An illustration of the SPAM attention module. 39

3.4 Positional Attention Module (PAM). 40

3.5 Sample images from the CoMoFoD dataset: (a) Real image, (b) Forged image with copy-move manipulation (c) Corresponding mask highlighting the manipulated regions, and (d) Binary mask indicating the exact forged areas. 43

3.6 Sample results of the proposed model on the (a) CASIA v2 (b) CoMoFoD, and (c) COVERAGE datasets. 46

3.7 Some error cases of the proposed model, where the red encircled region highlights the incorrect segmentation produced by the model. 48

4.1 The proposed ViXNet model extracts both local and global image features for deepfake video detection. It comprises two primary components: one that first extracts masked local features and then captures global inconsistencies among them, and another that generates global image features. The model consists of the following key modules: (A) a patch-wise self-attention module that generates local features by applying masking to each image patch, (B) a multi-headed self-attention module that detects global inconsistencies among the extracted local features, (C) a CNN backbone responsible for generating global image features, and (D) a feature stacking and classification module that combines intra- and inter-patch features from (B) with spatial global features from (C) to determine whether an image is real or fake. . . . 51

4.2 A single-headed self-attention unit in transformer blocks helps identify global correlations among patch elements. When multiple such units are stacked in parallel, they form multi-headed self-attention mechanisms. 53
4.3 The architecture of the Xception model is utilized to extract global feature representations from input cropped face images. 54
4.4 Examples of real and forged images (generated using face-swapping techniques) from the following datasets: (a) FF++, (b) CeDF, (c) DFID 55
4.5 Illustration of image data preparation from a video. 57
4.6 Performance of ViXNet across three datasets in the intra-dataset experimental setup. 60
4.7 Confusion matrices obtained from the intra-dataset experimental setup, where ViXNet is trained, validated, and tested on (a) DFID, (b) FF++, and (c) CeDF datasets. 60
4.8 Confusion matrices from different inter-dataset experimental setups. X_Y_Z denotes training on dataset X, validation on dataset Y, and evaluation on dataset Z. 62
4.9 Examples of correctly and incorrectly classified face images by the proposed model. The samples include correctly classified cases—(a) true positives and (b) true negatives—as well as misclassified instances—(c) false positives and (d) false negatives. 67
4.10 The proposed method consists of three stages: (A) Preprocessing, which extracts cropped face images from input videos using MTCNN-based face detection; (B) Feature Generation and Selection, where an HFS technique selects a near-optimal feature set; and (C) Training/Testing, where a classifier determines whether the input video is real or forged. . 68
4.11 Illustration of the handcrafted feature extraction process from a cropped face image. The second image from the left displays the label number $t \in 1, 2, \ldots, n$ assigned to each extracted patch. Daisy features, represented by $\overrightarrow{t}$, are extracted from each patch and used to compute similarity across different patches. Pairwise norm-2 distances (i.e., $|\overrightarrow{i} - \overrightarrow{j}|$, where $i \in 1, 2, \ldots, n-1$ and $i, j \in i+1, 2, \ldots, n$) are calculated and stored in lists named as $Row - i$. The length of each $Row - i$ is $n - i$. These lists are then stacked to form the final feature vector, whose length is given by $(n-1) + (n-2) + \cdots + 1 = \frac{n(n-1)}{2}$. In this case, with $n = 25$, the resulting feature vector has a length of 300. 69
4.12 Flowchart of the proposed hybrid feature selection technique. 74

4.13 The steps involved in generating feature vectors in the proposed HFS model are represented using solid-lined arrows, whereas the steps corresponding to the non-HFS stages are depicted with dashed-lined arrows. Feature_4 and Feature_6 represent the final near-optimal feature subsets obtained through the HFS and non-HFS approaches, respectively. . . . 76

4.14 Test accuracies of the HFS-based deepfake detection method across ten independent executions. 77

4.15 Training time as a function of the number of features for the CeDF dataset. 80

4.16 Comparison of accuracies for various SOTA FS methods on the CeDF and FF++ datasets. "No FS" refers to classification performed using the original features (i.e., Feature_5). 83

5.1 Key modules of the proposed method for detecting duplicate frames in manipulated videos. 89

5.2 An illustration of SSIM score differences, where a significant increase is observed at the copied positions, while remaining below M_C in other cases: (a) Frames 180–300 copied to 320–440, (b) Frames 185–220 copied to 60–95, and (c) Frames 135–170 copied to 40–75 and 215–250. 94

5.3 Localization of frame duplication for (a) Category 2, (b) Category 3, and (c) Category 4 videos. 96

5.4 Overview of the video frame deletion detection methodology. 97

5.5 Examples of MSE plots for real (top row) and manipulated (bottom row) videos. Frame deletion artifacts are reflected as inconsistencies in the curve shapes. 98

5.6 CNN architecture used in Algorithm 5.3 for detecting frame deletion in videos based on input MSE plots. 99

5.7 Second-order derivative curves of MSE values corresponding to the three previously analyzed manipulated videos, highlighting fluctuations attributed to frame deletion. 100

5.8 Example frame sequences for the actions: running, jogging, and walking. Each yellow-highlighted region denotes a distinct shot, and the corresponding frame indices are shown below each frame. The sequences include multiple shots. When consecutive shots are removed, the adjacent frames before and after the removed segment become continuous. For example, in the running sequence, frames 1 and 63 appear consecutively; in jogging, frames 4 and 52; and in walking, frames 14 and 100. This results in similar-looking frames appearing next to each other. . . 103
5.9 Confusion matrices for the best and worst results from five-fold cross-validation. (a) represents Fold-3 with the highest accuracy and F1 score, while (b) shows Fold-1 with comparatively lower performance. 104
5.10 Confusion matrix representing the classification performance of the CNN-based forgery detection module using the test set described in Section 5.4.6. 105
5.11 Bar chart showing the number of correctly localized deleted frames under various post-processing operations (BI, GB, BD, and GN). WPP represents the baseline scenario without post-processing, evaluated using Algorithm 5.4. 107
5.12 Visual comparison of MSE curve variations following different post-processing operations applied to the same set of fake videos depicted in Figure 5.5. Rows correspond to: (a) brightness increase, (b) brightness decrease, (c) Gaussian blur, and (d) Gaussian noise. 108
5.13 Failure analysis of the proposed method based on frame difference and second-order derivative curves. Various conditions such as noise, motion instability, and abnormal patterns are shown to affect detection and localization performance. 109

List of Tables

2.1 Comparison between active and passive forgery detection approaches . 21

3.1 CMFD results for different model architectural configurations based on standard metrics. All values, except for the number of parameters, are presented in %. 44

3.2 Performance evaluation of SPAM components using standard metrics. All values are presented in %. 44

3.3 CMFD results for different distance metrics used in SPAM based on standard evaluation metrics. All values are expressed in %. 45

3.4 Comparison of the proposed model's performance with SOTA methods on the CASIA v2 dataset under different protocols. A "-" indicates that no corresponding results were reported. All values are in %. 45

3.5 Comparison of the proposed model's performance with SOTA methods on the CoMoFoD dataset. A "-" indicates that no corresponding results were reported. All values are expressed as %. 47

3.6 Comparison of the proposed model's performance with SOTA methods on the COVERAGE dataset. A "-" indicates that no corresponding results were reported. All values are in %. 47

3.7 Performance comparison of the proposed model with SOTA methods on MICC-F600. A "-" indicates that no corresponding results are available. All values are expressed in %. 47

4.1 Performance of ViXNet and its component-based variations in an intra-dataset experimental setup, where the model is trained, validated, and tested on DFID. 60

4.2 Performance of ViXNet and its component-based variations under the inter-dataset experimental setup. The results correspond to models trained on DFID. 61

4.3 Performance of ViXNet under the inter-dataset experimental setup across different dataset configurations. 61

4.4 Test accuracy (in %) of ViXNet compared with various SOTA models under different experimental protocols. The experimental protocols follow the format Train Dataset_Validation Dataset_Test Dataset. M1 - M9 correspond to the methods proposed by Afchar et al. [42], Afchar et al. [42], Chollet et al. [47], Qian et al. [228], Guo et al. [107], Wodajo and Atnafu [35], Mohiuddin et al. [43], Ganguly et al. [44], and Coccomini et al. [229]., respectively. 63

4.5 Test AUC scores (in %) of ViXNet compared with SOTA models under various experimental protocols. Experimental protocols follow the format Train Dataset_Validation Dataset_Test Dataset. M1 - M9 correspond to the methods proposed by Afchar et al. [42], Afchar et al. [42], Chollet et al. [47], Qian et al. [228], Guo et al. [107], Wodajo and Atnafu [35], Mohiuddin et al. [43], Ganguly et al. [44], and Coccomini et al. [229]., respectively. 64

4.6 Comparison of model performance on the DFDC dataset. Methods marked with * were evaluated under the experimental setup described in this work. 66

4.7 Performance of the HFS model and its components on the CeDF dataset (see Figure 4.13). The results under Feature_4 and Feature_6 correspond to HFS-based and non-HFS-based feature selection, respectively. Max, Min, Avg, and SD denote maximum, minimum, average, and standard deviation, respectively. 78

4.8 Performance of the HFS model and its components on the FF++ dataset (see Figure 4.13). The results under Feature_4 and Feature_6 correspond to HFS-based and non-HFS-based feature selection, respectively. Max, Min, Avg, and SD denote maximum, minimum, average, and standard deviation, respectively. 79

4.9 Performance evaluation when HFS is applied to one dataset to obtain the near-optimal feature subset and the indices of selected features, which are then used to extract only the selected features from another dataset for classification. Case 1 represents the scenario where HFS is applied to FF++ to obtain feature indices, and classification is performed using the selected features extracted from CeDF. Conversely, Case 2 represents the scenario where HFS is applied to CeDF, and classification is conducted using the selected features extracted from FF++. 81

4.10 Performance of the proposed model on a inter-dataset experimental setup. 82

4.11 Performance of the proposed model in the intra-dataset experimental setup. 84
4.12 Performance of the proposed HFS model and its components (see Figure 4.13) on DFDC dataset. 84
4.13 Comparison of model performance on the DFDC dataset. Methods marked with * were evaluated under the experimental setup described in this work. 85

5.1 The current dataset includes forged videos with frame duplication. The first 10 videos contain single-instance frame duplication, while the last three exhibit multiple-instance frame duplication. 94
5.2 Presents both category-wise and overall performance of the proposed frame duplication detection technique. 95
5.3 Duplicate frame detection results for the videos with static and non-static background 95
5.4 Presents the frame duplication detection performance on videos affected by distortion operations such as noise addition, vintage effects, and contrast changes. 96
5.5 Performance results of frame deletion detection using standard evaluation metrics. Results are presented as mean (x) ± standard deviation (y). 104
5.6 Detection performance of the proposed method under different post-processing conditions. The evaluation metrics are presented as mean ± standard deviation across 5-fold cross-validation. BI, BD, GB, and GN refer to brightness increase ($\alpha = 1.10$, $\beta = 20$, Factor = +10%), brightness decrease ($\alpha = 0.90$, $\beta = -20$, Factor = -10%) , Gaussian blurring (Kernel $(5, 5)$, $\sigma = 1.0$), and Gaussian noise ($\sigma = 4$), respectively. 107

List of Acronyms/Abbreviations

2D	Two Dimensional
3D	Three Dimensional
acc	accuracy
AI	Artificial Intelligence
AUC	Area Under the receiver-operating characteristic Curve
CBAM	Convolutional Block Attention Module
CeDF	Celeb-DF (V2)
CMFD	Copy-move Forgery Detection
DA	Detection Accuracy
DCT	Discrete Cosine Transform
DFDC	DeepFake Detection Challenge
DFID	Deepfake Image Dataset
DWT	Discrete Wavelet Transform
FF++	FaceForensics++
FN	False Negative
FP	False Positive
FS	Feature Selection
GANs	Generative Adversarial Networks
GLCM	Gray Level Co-occurrence Matrix
GOP	Group of Pictures
GWO	Grey Wolf Optimization
HFS	Hierarchical Feature Selection
HOG	Histogram of Oriented Gradients
HSV	Hue-Saturation-Value
LBP	Local Binary Pattern
LCSC	Longest Common Subsequence Comparison
ML	Machine Learning
MRB	MultiRes blocks
MSE	Mean Squared Error
MTCNN	Multi-Task Cascaded Convolutional Networks
PAM	Positional Attention Module

PCA	Principal Component Analysis
PSO	Particle Swarm Optimization
PWSA	Patch-Wise Self-Attention module
RANSAC	Random Sample Consensus
RB	Residual blocks
ReLU	Rectified Linear Unit
SIFT	Scale-Invariant Feature Transform
SOTA	State-of-the-Art
SPAM	Similarity-based Positional Attention Module
SSIM	Structural Similarity Index Measure
SURF	Speeded-Up Robust Features
SVM	Support Vector Machine
TN	True Negative
TNR	True Negative Rate
TP	True Positive
TPR	True Positive Rate
VS	Vortex Search

Chapter 1

Introduction

In the digital age, images and videos serve as powerful mediums for communication, documentation, education, entertainment, and surveillance. With the proliferation of smartphones, digital cameras, and social media platforms, multimedia content has become an integral part of daily life. The increasing reliance on digital visual information has also amplified concerns regarding its authenticity [1]. Manipulated images and videos can easily mislead viewers, distort facts, and influence public opinion. These concerns have given rise to a critical area of research: image and video forgery detection [2–4].

Digital forgeries refer to the intentional modification or synthesis of visual content to misrepresent reality. While digital editing has long been used for creative and practical purposes, such as in advertising, film production, and photography -the line between legitimate editing and deceptive forgery has become increasingly difficult to define. In its earliest form, image manipulation was carried out using manual editing tools like Adobe Photoshop, requiring considerable time, effort, and expertise. Basic operations such as cropping, cloning, retouching, or combining elements from multiple images (splicing) could be used to alter the narrative conveyed by an image [5]. In video editing, techniques such as frame removal, duplication, or temporal rearrangement were similarly used to obscure or fabricate events [6].

However, the landscape of forgery has transformed dramatically with the advent of artificial intelligence (AI) and deep learning. These technologies have enabled the automatic generation and alteration of media with minimal human intervention. Particularly influential in this regard are Generative Adversarial Networks (GANs) [7], which can synthesize realistic images and videos from scratch. GANs work through an adversarial training process where a generator attempts to create convincing forgeries, while a discriminator tries to differentiate them from real samples. As training progresses, the generator improves to the point where even humans struggle to distinguish real content from fake. This capability has led to the development of highly realistic synthetic media, popularly referred to as *"deepfake"*. This poses significant social risks, as a single manipulated video can distort truth and mislead audiences. For example, the "Synthesizing Obama" project (2017) modified video footage (see Figure 1.1 of former USA President Barack Obama to make it appear as though he

(a) Real **(b)** AI generated

Figure 1.1: (a) Real video showing an unaltered, authentic clip of former President Barack Obama. (b) Fake video generated by AI software that creates highly realistic deepfake footage of Obama using existing audio and video clips [9].

spoke entirely different words from a separate audio track [8].

Modern forgery techniques extend beyond traditional cut-and-paste operations and now include subtle and seamless modifications that preserve lighting, expressions, and motion. These techniques can generate synthetic faces, manipulate lip movements to match altered audio, or even recreate full-body animations of individuals. Such forgeries pose severe risks in areas like journalism, law enforcement, national security, and personal privacy. For instance, a falsified video can mislead legal investigations, incite public unrest, or damage reputations.

The growing accessibility of forgery tools has exacerbated these threats. Mobile applications such as FaceApp[1], ZAO, and Reface, as well as open-source platforms like DeepFaceLab[2] and FaceSwap [3], have made it easy for virtually anyone to create manipulated content. While many of these applications are used for entertainment or satire, their underlying technology can be exploited for malicious purposes such as misinformation campaigns, identity theft, and cyberbullying.

In parallel, the detection of image and video forgery has become a pressing challenge. Conventional forensic techniques that rely on visual inspection or metadata analysis are often insufficient against advanced manipulations. This has led to the emergence of *passive* and *active* forgery detection methods [10]. *Passive* methods analyze intrinsic features of media, such as inconsistencies in pixel-level statistics, sensor

[1]https://apps.apple.com/gb/app/faceapp-ai-face-editor/id1180884341
[2]https://apps.apple.com/us/app/deepfacelab-face-swap-editor/id1568914185
[3]https://github.com/MarekKowalski/FaceSwap

noise patterns, or compression artifacts. These methods do not require prior knowledge of the original content. *Active* methods, in contrast, involve embedding digital watermarks or signatures during the creation of content to verify authenticity later. Both approaches are under continuous development as forgers devise increasingly sophisticated methods to evade detection.

As the arms race between forgery creation and detection intensifies, the research community is focusing on developing more robust and generalizable detection algorithms. These include machine learning (ML) and deep learning-based classifiers, attention mechanisms, and multimodal analysis that combines visual, audio, and contextual cues. However, challenges such as adversarial attacks, generalization across forgery types, and real-time detection still remain open areas of investigation. Overall, the manipulation of digital images and videos has evolved from manual editing to highly automated and realistic synthetic content generation. This evolution has introduced significant concerns across multiple domains, prompting the need for effective forgery detection techniques. Understanding the underlying principles, existing approaches, and current limitations is essential to developing robust and trustworthy digital media verification systems. This study aims to explore, analyze, and compare various image and video forgery detection techniques, contributing to the growing effort to preserve the integrity of digital visual content.

1.1 Importance of Digital Media Integrity

In an era where digital media is a primary means of communication, information dissemination, and documentation, ensuring its integrity has become more critical than ever. Digital images and videos serve as evidence in legal cases, tools for journalism, means of social interaction, and records of historical events. However, the increasing accessibility of sophisticated editing tools and AI-driven manipulation techniques has raised serious concerns about the authenticity of digital content [11, 12]. When manipulated media spreads unchecked, it can mislead the public, fuel misinformation campaigns, and undermine trust in reliable sources. The integrity of digital media is essential to maintaining credibility in journalism, ensuring justice in legal proceedings, and preserving security in national and international affairs.

Forensic investigations, law enforcement agencies, and cybersecurity experts rely on digital media as critical evidence to solve crimes, detect fraud, and authenticate identity. If manipulated media is introduced as genuine, it can lead to wrongful accusations, financial fraud, and misinformation-driven conflicts. Similarly, in journalism, media authenticity is crucial in upholding public trust. When fake or manipulated content circulates, it erodes confidence in legitimate news sources, leading to skepticism

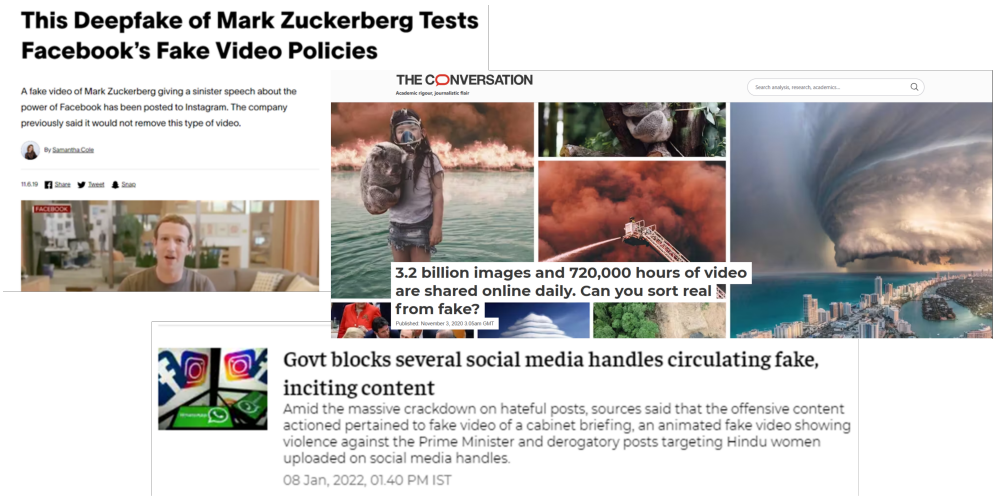


Figure 1.2: Snapshots of news articles highlighting the societal impact and risks associated with manipulated media, reflecting growing public and academic concern.

and confusion. In security and defense, forged media can be weaponized for propaganda, cyber warfare, and espionage, creating false narratives that influence public perception and political stability. Furthermore, with the growing reliance on biometric verification in banking, immigration, and online security, forged media can enable identity theft and unauthorized access to sensitive systems.

Maintaining digital media integrity is not just about detecting and preventing forgery but also about fostering responsible use of technology. Ethical considerations in AI and ML play a vital role in ensuring that advanced image and video generation techniques are not misused. Governments, researchers, and companies must work collaboratively to develop robust forgery detection methods while implementing policies that promote ethical standards in digital content creation. The significance of digital media integrity extends beyond technology; it is a foundation for trust, security, and ethical communication in a world increasingly dependent on digital interactions.

1.2 Rise of Image and Video Forgeries

The rise of image and video forgeries is closely tied to the explosive growth of digital media in modern society. The creation and sharing of multimedia content in the social media platforms have become integral to daily life with the widespread availability of smartphones, and high-resolution cameras. This is because the visual content has gained dominance in how people communicate and consume information. Accordingly, people gained incentive to manipulate it, whether for entertainment, satire, political gain, or malicious purposes [13]. Figure 1.2 illustrates this growing concern by showing snapshots of various news articles highlighting the impact and risks associated with manipulated media.

Historically, image manipulation can be traced back to the analog era, when techniques such as airbrushing and darkroom compositing were used to alter photographs. These methods were time-consuming and required specialized expertise. The digital revolution, however, significantly lowered the barrier to media editing. Software such as Adobe Photoshop made it possible for users to edit images with ease, and over time, even non-specialists have gained access to tools that can produce convincing results [14]. However, the widespread accessibility of editing tools has contributed to a dramatic increase in the frequency and sophistication of manipulated media. What was once limited to professionals is now achievable by anyone with a smartphone and access to editing apps. This democratization of forgery tools has led to the proliferation of manipulated content across the internet and social platforms [15].

These forgeries are no longer limited to crude alterations or basic copy-paste. They have become more subtle and often indistinguishable from authentic media, especially when shared in low-resolution formats or among fast-moving information streams. Forged content is now commonly used to mislead audiences, influence opinions, or even distort historical and political narratives [6]. From altered celebrity images to fabricated news videos, manipulated media is regularly encountered in both casual and professional digital environments [16].

In the context of video forgery, one particularly deceptive class involves inter-frame manipulations. These include operations such as frame insertion, frame deletion, and frame duplication that are capable of subtly altering the sequence of events portrayed in a video. Unlike traditional deepfake manipulations, which often target the visual appearance or expressions of individuals, inter-frame forgeries exploit the temporal dimension of video content. For example, removing a few frames from a surveillance clip could obscure critical actions, while duplicating frames may be used to exaggerate or fabricate behavior. Such manipulations can be extremely difficult to detect through visual inspection alone, especially when the edits are performed in a way that preserves the video's overall continuity and quality [17].

The implications of these forgeries extend beyond technology. In a time when visual information is often equated with factual evidence, the existence of sophisticated forgeries undermines public trust in digital media. This erosion of credibility can foster misinformation, extend societal divisions, and complicate legal processes that depend on visual documentation [17]. Furthermore, the speed at which forged media spreads online amplifies its impact. Manipulated content can go viral within minutes, often reaching wide audiences before any form of verification or correction can take place. Platforms such as Twitter, WhatsApp, and TikTok enable rapid sharing without built-in authenticity checks, giving forged media a dangerous window to influence public

perception [18].

As a result, the need for effective detection strategies is growing in every passing days. Combating this growing threat involves not only technological advancements in forgery detection but also fostering a culture of critical media consumption and accountability. Interdisciplinary efforts that combine computer science, forensics, law, and public policy are essential to counteract the evolving nature of digital media manipulation [19].

1.3 Classification of Image Forgery Techniques

The common saying "seeing is believing" is no longer valid in the digital era, where the reliability of digital images is increasingly at risk. In today's world, verifying digital media before accepting it as truth is essential. However, media forgery is not a recent issue it has existed for decades. Numerous historical examples demonstrate early instances of image manipulation. Figure 1.3 a famous doctored portrait of Abraham Lincoln features his head on John C. Calhoun's body, with altered text to enhance Lincoln's legacy. Image manipulation was common during the Civil War, shaping public perception. This forgery became iconic, with its origins revealed in 1970 by Library of Congress archivist Milton Kaplan. Figure 1.4 David King's The Commissar Vanishes exposes how Stalin erased executed rivals from photos during the 1930s purges. Fearing punishment, citizens also removed banned individuals. King discovered these manipulations in 1970, revealing Soviet propaganda's impact on history. Figure 1.5a and Figure 1.5b, these images show the original and altered versions of a 2008 Iranian missile test. The doctored image, published by an official media journal, falsely depicts four missiles launching instead of three. The fourth missile was digitally copied and pasted from an existing one to create a misleading impression of increased military capability.

Image forgery is the process of altering an image to mislead viewers. It is commonly used in misinformation, propaganda, and digital fraud. There are three main types of image forgery [23]: copy-move, splicing, and image retouching. These categories represent the foundation of most tampering techniques used in digital media. The hierarchical classification of image forgery types is presented in Figure 1.6, providing a structured overview of the forgery categories. A concise overview of the main categories of image forgery is provided below:

1. **Copy-Move Forgery**: This technique involves duplicating a part of an image and placing it elsewhere within the same image to conceal or replicate an object. Since the copied section originates from the same image, factors like lighting, noise, and texture remain consistent, making detection difficult (see Figure 1.5).

Figure 1.3: Lincoln's face from the five-dollar bill photo was flipped and placed onto a slavery proponent's body from an 1852 engraving [20].

Figure 1.4: "The Commissar Vanishes", David King's visual history of the falsification of images, explores how Stalin manipulated photography to erase all memory of his victims [21].

(a) **(b)**

Figure 1.5: (a) Real Image: The unaltered, authentic photograph, (b) Fake Image: A digitally manipulated version of the original [22].

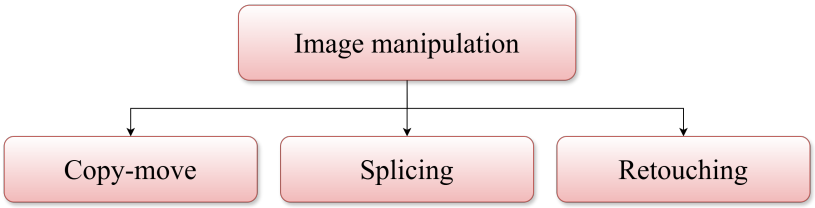


Figure 1.6: Classification of image forgery techniques.

Common examples include artificially increasing a crowd size in political rallies or removing objects like watermarks by covering them with copied background elements.

2. **Splicing**: Splicing merges elements from different images to fabricate a misleading composite. This method is often used to create false scenarios that never actually happened. Notable examples include the doctored portrait of Abraham Lincoln, where his head was placed on John C. Calhoun's body (see Figure 1.3), and manipulated celebrity photos where faces are swapped to create fake scandals.

3. **Retouching**: Retouching modifies an image to enhance or alter its appearance. While commonly used in fashion and media, it can also serve deceptive purposes, such as distorting historical records or spreading propaganda. Examples include Stalin's regime erasing political enemies from photographs (see Figure 1.4) and excessive airbrushing in magazines to promote unrealistic beauty standards.

1.4 Classification of Video Forgery Techniques

Video tampering refers to the process of altering video content to produce forged or manipulated footage. It can be broadly categorized into three major types [24]: (i) intra-frame (spatial tampering) forgery [25, 26], (ii) inter-frame (temporal tampering) forgery [27, 28], and (iii) spatio-temporal forgery [29, 30]. Each category uses different techniques to alter the authenticity of the video content, either by modifying specific frames, manipulating the sequence of frames, or combining both methods for a more complex form of deception. The hierarchical classification of video forgery techniques is illustrated in Figure 1.7, which organizes the various forms based on their operational characteristics. Below is a brief description of the major types of video forgery:

1. **Intra-frame Forgery**: In intra-frame forgery, spatial tampering can occur at either the pixel level or block level across selected frames or throughout the video. This process involves adding, removing, or modifying entire objects or

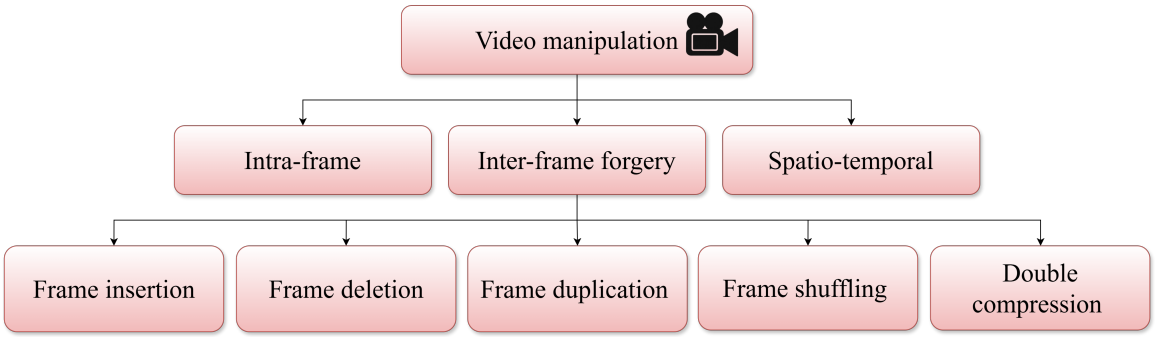


Figure 1.7: Hierarchical classification of video forgery techniques.

(a) **(b)**

Figure 1.8: General types of intra-frame video forgeries (a) original frame, (b) fake frame [31].

specific parts of objects within individual frames. These manipulated elements can belong to either the foreground, such as a person or moving object, or the background, like scenery or static structures. Such alterations are designed to mislead viewers by changing the visual content without affecting the sequence of frames [26]. Advanced editing tools can make these changes appear seamless, making detection challenging. This form of forgery is commonly used in scenarios where specific elements need to be hidden, highlighted, or fabricated to convey a false narrative. A clear illustration of intra-frame forgery is shown in Figure 1.8, where a lamppost present in the original video has been digitally removed from each frame. The manipulation is performed with high precision to maintain spatial consistency, making the forgery visually undetectable to casual observers.

Another example is upscale crop forgery [32] which is used to hide evidence of criminal activity or remove unwanted objects near the frame boundaries of a video. For instance, if frames 40–70 in a video of spatial resolution 600×800 show sensitive content at the edges, they can be enlarged and then center-cropped back to the original size, effectively removing boundary details. These modified frames are then reinserted into the video to maintain visual consistency. This

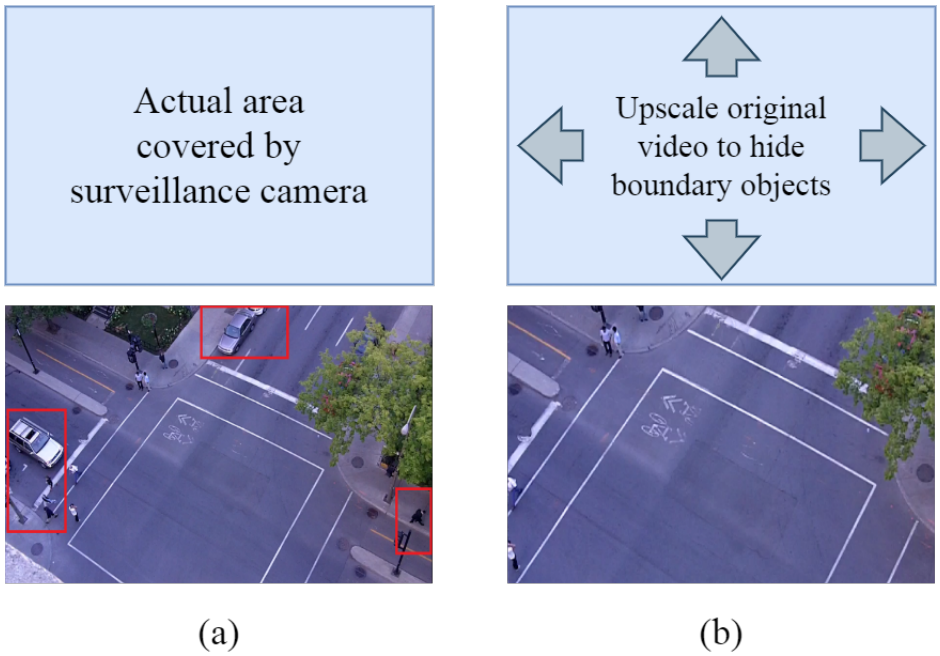


Figure 1.9: Upscale video forgery process (top row): (a) original video frame, (b) forged video frame created by upscaling the original to conceal boundary areas (highlighted in red).

technique ensures the background remains unchanged while concealing specific content. Figure 1.9 demonstrates the upscale forgery generation process with an example.

2. **Inter-frame Forgery**: In inter-frame tampering, manipulation occurs at the frame level, where extra frames are added or existing ones are removed to alter the presence or actions of one or more objects. Additionally, the sequence of frames can be rearranged, which changes the timing and flow of events in the video. Rather than modifying the visual content within individual frames, this type of forgery focuses on altering the order, duration, or continuity of frames to distort the actual sequence of events. Common techniques include deleting frames to conceal specific actions, inserting duplicate frames to extend the appearance of certain events, or rearranging frames to misrepresent the true order of occurrences. For instance, critical moments in a surveillance video could be removed to hide evidence of illegal activity, or the sequence of events could be shuffled to create a misleading narrative. This manipulation can significantly affect the interpretation of the video, making it an effective yet deceptive form of tampering. Figure 1.10 illustrates the general types of inter-frame video forgeries. These include frame insertion from another video, frame deletion, and frame duplication (video copy-move forgery).

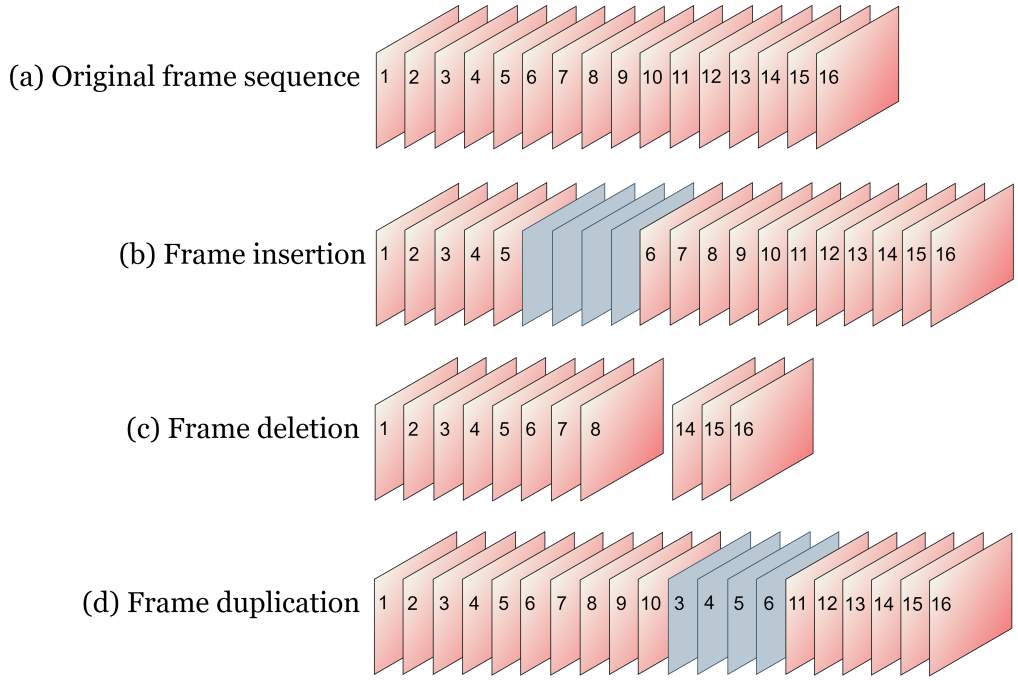


Figure 1.10: General types of inter-frame video forgeries. (a) Original frame sequence, (b) Frame insertion from another video, (c) Frame deletion, and (d) Frame duplication (video copy-move forgery).

3. **Spatio-temporal Forgery**: Spatio-temporal tampering is a combination of both intra-frame (spatial) and inter-frame (temporal) manipulations, making it the most complex and sophisticated form of video forgery. This method involves altering the visual content within individual frames while simultaneously modifying the sequence or timing of those frames. By combining these two techniques, highly convincing fabrications can be created objects or individuals may be added, removed, or modified within specific frames (spatial manipulation), while the order or duration of frames is adjusted to distort the timeline of events (temporal manipulation). For example, deepfake video forgeries [33–35], where faces are artificially altered fall under this category, as they often involve both spatial and temporal adjustments to create realistic yet deceptive video content. Figure 1.11 illustrates some real faces and their corresponding Deeepfake faces.

1.5 Problem Definition: Challenges Posed by Various Types of Forgeries

The increasing sophistication of digital image and video manipulation techniques present significant challenges in maintaining media authenticity. Various types of forgeries, such as image copy-move, deepfake, and frame duplication, pose distinct

Figure 1.11: Examples of real facial images alongside their corresponding Deepfake versions.

threats across different domains, including journalism, forensics, security, and digital identity verification. The ability to alter, replicate, or synthesize media content with high precision makes detection a complex task, often requiring advanced computational techniques and forensic analysis. As forgeries become more refined, traditional detection methods struggle to keep pace, necessitating the development of more robust and adaptive forgery detection mechanisms.

One of the most prevalent forms of image forgery is *copy-move forgery*, where a part of an image is duplicated and pasted within the same image to obscure or alter specific details. This technique is commonly used to remove objects, hide critical information, or falsify evidence. Since the copied region originates from the same image, maintaining similar lighting conditions and textures, detecting these manipulations becomes highly challenging. Traditional detection approaches based on pixel-based analysis often fail when the copied region undergoes transformations such as rotation, scaling, or compression. Advanced techniques leveraging deep learning and feature extraction are required to accurately identify tampered regions and distinguish them from genuine content.

Deepfake technology, driven by AI, represents one of the most alarming advancements in media forgery. Using deep learning models such as GANs and autoencoders, deepfake can seamlessly replace a person's face in a video, modify facial expressions, and even generate entirely fabricated speech. These hyper-realistic forgeries pose significant risks, particularly in misinformation campaigns, political propaganda, and cybercrime. Unlike traditional forgeries, deepfake are highly resistant to conventional forensic techniques, as they generate visually plausible content with minimal detectable artifacts. The evolving nature of deepfake algorithms demands continuous improvements in detection methodologies, including deep learning-based classification models,

temporal consistency analysis, and biometric feature examination.

Another critical forgery type affecting video content is *frame duplication forgery*, where specific frames in a video sequence are duplicated, removed, or shuffled to alter the context or misrepresent an event. This manipulation is commonly used in surveillance footage tampering, misleading news broadcasts, and deceptive video evidence. Detecting duplicate frame forgeries is particularly difficult in long-duration videos, where subtle changes might go unnoticed. Traditional frame-by-frame analysis is computationally expensive and often ineffective against sophisticated frame alterations. Advanced detection approaches, such as sequence matching, motion analysis, and neural network (NN) based temporal modeling, are necessary to identify inconsistencies and detect manipulated sequences.

The challenges posed by these forgery techniques highlight the urgent need for robust, automated, and scalable detection mechanisms. The increasing complexity of forgeries, coupled with their widespread misuse, raises serious ethical, legal, and security concerns. All detection mechanisms fall under either *active* or *passive* detection approaches, which will be discussed in Chapter 2 in details. *Active* detection relies on predefined information, such as watermarks or digital signatures, and the presence or absence of this information directly determines the success of tampering identification. However, such auxiliary data is often unavailable or inaccessible in practical scenarios. Hence, the present thesis follows *passive* detection methods, which do not require any prior embedded information and instead focus on analyzing the inherent inconsistencies within the media content itself. This makes passive methods more suitable for detecting forgeries in real-world applications where no additional authentication data is provided. Addressing these challenges requires an interdisciplinary approach, combining ML, computer vision, and forensic analysis to develop effective forgery detection frameworks. By improving detection accuracy and efficiency, researchers can contribute to safeguarding digital media integrity and mitigating the risks associated with manipulated content.

1.6 Objectives of the Thesis

The present research aims to achieve the following objectives.

- To conduct an in-depth literature survey on image and video forgery detection techniques, identify the limitations of existing methods, and utilize these findings to develop effective detection approaches tailored to various types of forgeries.

- To develop lightweight and robust forgery detection methods capable of real-time performance, including the localization and identification of copy-move forgeries

in digital images, even under various post-processing operations.

- To detect deepfake in images and videos by implementing deep learning-based models, including Convolutional Neural Networks (CNNs) and transformer-based architectures, and exploring the effectiveness of combining deep features with handcrafted features to enhance detection performance.

- To develop methods for detecting inter-frame video forgery techniques, such as frame duplication and frame deletion, by identifying temporal inconsistencies without the need for the original video.

1.7 Scope of the Thesis

The work presented in this thesis addresses the challenges associated with the analysis of image and video forgery detection techniques. Initially, the primary objective is to explore and examine a wide range of existing forgery detection methods. To achieve this, an extensive literature review has been conducted, covering various approaches used in image and video forgery detection [6]. This detailed literature study covers traditional handcrafted feature-based methods, which rely on pixel-level or block-level inconsistencies; statistical approaches that focus on noise patterns and compression artifacts; and deep learning models such as CNNs and transformer-based architectures, which learn high-level semantic features. Special attention is given to hybrid methods that combine deep features with handcrafted features to improve detection performance and robustness. Through this analysis of various forgery techniques, certain image and video forgery techniques are identified as particularly concerning due to their significant impact on society and their widespread use on social media platforms [36, 37]. Based on the findings, the study focuses on three major types of image and video forgeries: (i) Copy-move forgery in images [37,38], where a part of an image is copied and pasted elsewhere within the same image; (ii) Deepfake manipulations in images and videos [33,39], which are generated using deep generative models like GANs and autoencoders; and (iii) Inter-frame video forgeries [4, 40], such as frame duplication and frame deletion, which introduce temporal inconsistencies in video sequences. Through this analysis, several limitations of existing forgery detection methods have been identified, and new techniques have been developed specifically to address these shortcomings.

To detect image copy-move forgeries, a customized MultiResUNet [41] architecture is employed for copy-move forgery detection (CMFD). Leveraging multi-scale feature extraction, the model captures both fine and high-level visual patterns essential

for identifying duplicated regions. Forgery localization is achieved by precisely highlighting manipulated areas within the image. To enhance computational efficiency, standard convolutions in MultiRes blocks are replaced with separable convolutions, reducing trainable parameters while maintaining performance making the model suitable for real-time use. Residual connections are incorporated to address the vanishing gradient problem and ensure stable training, while preserving spatial details through better encoder-decoder information flow. The model's robustness is evaluated on various post-processed datasets, demonstrating effectiveness under compression and geometric transformations, ensuring reliable performance across diverse forgery scenarios.

In the early phase of this thesis, various approaches are explored to enhance deepfake detection in both images and videos by focusing on manipulation artifacts. While existing methods often target facial regions like the eyes, nose, and lips, this study emphasizes a composite feature representation by extracting both global and local features. The face image is segmented into patches to capture local inconsistencies, while the entire face is used for global feature extraction. An initial experiment evaluates the impact of color models: RGB, HSV, and YCbCr—by passing images through separate Mesoinception-4 [42] networks and averaging the extracted features to assess the influence of color on detection accuracy [43]. A more advanced method uses Xception to extract global features from cropped face images, applying a soft attention mechanism to highlight local artifacts, with attention-weighted features classified through dense layers [44]. Later, local and global features are fused using two methods: ViXNet [45] and a Hierarchical Feature Selection approach [46], which remove redundant features to retain the most informative ones. Finally, stacking-based ensemble strategies are introduced, where features from Xception Network [47] and EfficientNet-B7 are combined, ranked, and refined before classification via a multi-layer perceptron, enhancing detection robustness [48, 49].

The first method addresses the limitations of existing approaches that focus only on specific facial regions. It expands the region of interest to the entire face by combining both local and global features, thereby increasing the receptive field and enhancing the model's ability to detect manipulations more robustly. A hybrid architecture, integrating a Transformer with the Xception Network is employed to learn the consistency of subtle artifacts left by deepfake generation methods across the facial region. This approach contains two branches: one branch is used to capture inconsistencies in local facial regions through a patch-wise self-attention module and transformer, while the other branch is used to extract global spatial features using a deep CNN.

Additionally, deep feature-based methods often generate a large number of features, many of which may be redundant or irrelevant, leading to increased computational

cost. These limitations highlight the need for effective feature selection by analyzing and eliminating unnecessary features. In addition, to check local inconsistencies, handcrafted features are generated in place of one path in the previous architecture specifically, the branch that captures inconsistencies in local facial regions using a patch-wise self-attention module and transformer. The other path of previous model, which extracts global features from the full face image, remains intact. Here, the effectiveness of combining deep features with handcrafted features is explored through a hierarchical feature selection approach. Both types of features are extracted and passed through a selection process where a hybrid feature selection technique is used. This technique combines Grey Wolf Optimization (GWO) [50] and the Vortex Search (VS) [51] algorithm to select an optimal feature set.

At the early stage of inter-frame video forgery detection, an ensemble-based technique [52] is proposed for detecting frame duplication in videos (also known as copy-move), where the results from three distinct detectors are combined using a majority voting strategy. Each base detector is designed using a fixed detection strategy but different texture-based features, including two variants of the Gray Level Co-occurrence Matrix (GLCM) and the Local Binary Pattern (LBP) descriptor. The method begins by extracting features from each frame and sorting them lexicographically to group similar frames together, facilitating the identification of duplicate sequences. To address the issue of fractional detection where only parts of duplicated sequences are detected—a post-processing step is introduced to merge partially detected segments into complete duplicated sequences when necessary. This approach is capable of detecting both single and multiple instances of copy-move forgeries within a video and is robust against common postprocessing techniques used by forgers, such as adjustments in brightness, contrast, blurring, and the addition of noise. The proposed method was validated on an in-house dataset of 300 videos and demonstrated superior performance compared to several state-of-the-art (SOTA) techniques.

Subsequently, to challenge the address of detecting inter-frame video forgery, particularly frame duplication, a statistical-based method is developed that identifies temporal inconsistencies without requiring access to the original video. The method aims to work effectively in both static and dynamic background scenarios, where duplicated frames may not be visually distinguishable. For this purpose, the Structural Similarity Index Measure (SSIM) [53] is used to compute the similarity between consecutive video frames. SSIM serves as a reliable statistical measure to capture subtle temporal inconsistencies introduced by frame duplication. A simple yet effective searching algorithm is then applied to analyze the SSIM values and detect duplicated segments. This enables both detection and localization of tampered frames, making the method suit-

able for real-world scenarios where forged videos may undergo further post-processing or involve complex motion. In continuation, a frame duplication detection, a frame deletion detection method is developed specifically for surveillance videos using a two-step detection process. In the first step, the input video is preprocessed, and the Mean Squared Error (MSE) differences between consecutive frames are calculated to capture temporal inconsistencies. These differences are used to generate an MSE curve, which transforms the temporal information into a spatial representation. This curve is then fed into a CNN that is trained to detect features associated with frame deletion. If the video is identified as tampered, the method proceeds to a second step, where a second-order derivative analysis is applied to the MSE curve. This analysis helps confirm the presence of forgery by identifying sharp variations that typically indicate the positions of deleted frames.

1.8 Organization of the Thesis

This thesis provides a comprehensive study on image and video forgery detection techniques, with an emphasis on both classical approaches and modern deep learning-based methods. The goal is to investigate, implement, and evaluate various forgery detection strategies to address the growing threat of visual misinformation. In light of the overview, the thesis has been organized into the following chapters, each addressing specific aspects of image and video forgery detection:

i) **Chapter 2** provides a comprehensive overview of forgery types and detection techniques, establishing essential background knowledge for the thesis. It reviews both traditional and deep learning-based approaches for detecting image and video forgeries. Key challenges such as dataset limitations, robustness of detection models are also discussed to set the stage for the methods proposed in later chapters.

ii) **Chapter 3** focuses on CMFD in digital images. It presents a deep learning-based model specifically developed in this work to detect duplicated regions, which are often challenging to identify due to post-processing or transformations. The chapter discusses the dataset used, model architecture, training strategy, and performance evaluation. Results demonstrate the model's effectiveness in handling various copy-move manipulation scenarios.

iii) **Chapter 4** delves into the detection of deepfake forgeries in both images and videos. It examines the application of CNNs, transformers, and hybrid architectures to detect subtle facial manipulations commonly found in synthetic media.

This chapter highlights the design and evaluation of a custom model using attention mechanisms and multi-level feature extraction. In addition to this, feature selection is performed on a combination of handcrafted and deep features to investigate whether all CNN-based features contribute effectively to deepfake detection. Performance is validated on benchmark datasets, and results are compared with existing SOTA methods.

iv) **Chapter 5** addresses inter-frame video forgeries, including frame deletion and duplication attacks that alter the temporal structure of videos. It presents techniques for detecting these manipulations using statistical features and temporal analysis. Self-curated datasets are used to validate the proposed methods, and experimental results show high accuracy in identifying tampered segments. This chapter also emphasizes the need for temporal consistency analysis in video forgery detection.

v) **Chapter 6** consolidates the findings from all the previous chapters and discusses the strengths and limitations of the proposed methods. It highlights the importance of developing robust, real-time, and generalizable forgery detection frameworks for practical deployment. The chapter also outlines potential directions for future work, including real-time processing, cross-dataset evaluation, and multi-modal forgery detection.

Chapter 2

Literature Review

The chapter 1 presented a classification of image and video forgery techniques. The detection of such manipulated digital content is broadly categorized into two main approaches: active forgery detection [54, 55], and passive forgery detection [56–58]. Figure 2.1 illustrates this categorization in a hierarchical structure and a comparative overview is presented in Table 2.1 for better comprehension. *Active forgery* detection techniques depend on prior information embedded within the digital media, such as digital watermarks [59, 60] or source camera-based authentication [61, 62] data. These methods involve inserting identifying information, including watermarks or digital signatures, either during the content creation process or afterward using specialized software. This embedded information is later used to verify the originality, authenticity, and integrity of the content. Methods like, Singh and Singh [63] used Discrete Cosine Transform (DCT) based self-recoverable fragile watermarking scheme, generating authentication and recovery bits from the Most Significant Bits (MSBs) of each 2x2 pixels block. Authentication bits were embedded in the block's own Least Significant Bits (LSBs), while recovery bits were placed in mapped blocks. This two-level hierarchical detection technique improved tamper localization accuracy. Bansal and Mathuria [64] combined Discrete Wavelet Transform (DWT) and DCT for watermarking, embedding the watermark in the red color channel using DWT, followed by DCT on 8×8 blocks, which made the image more secure and robust. Information embedded at the Group of Pictures (GOP) or frame level has also been used in videos to enable further authentication. Tanima et al. [65] embedded an invisible bipolar watermark in the luminance of predictive P-frames in videos using a pseudorandom key, enhancing security and minimizing bitrate increase. Peak Signal-to-Noise Ratio (PSNR) was used to evaluate the visual quality and assess the impact of watermarking on the video. Navajit et al. [66] proposed a method to detect forgery in videos using wavelet transform coefficients to generate signatures for spatiotemporal tampering detection. A video's GOP is verified by comparing authentication codes; if the distance is below a set threshold, the GOP is considered authentic.

Source camera-related artifacts, also referred to as camera-specific signatures, have been utilized to detect image and video tampering by analyzing unique patterns introduced during the capture process. These artifacts aid in verifying the authenticity

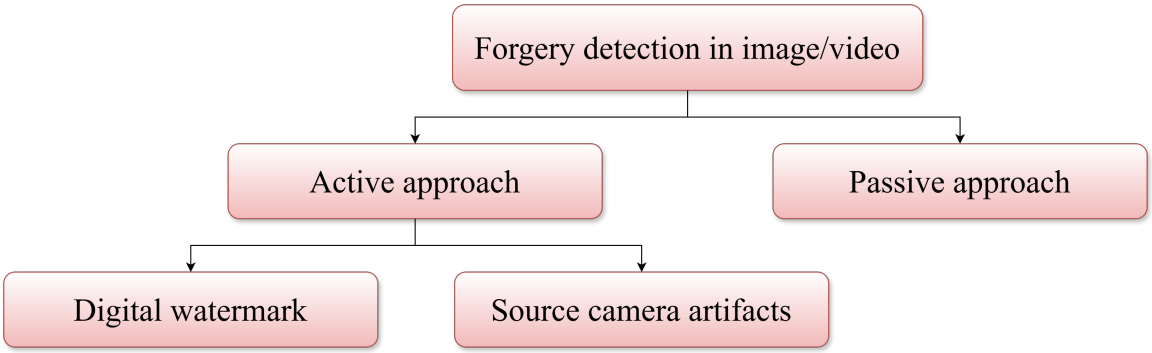


Figure 2.1: Classification of image/video forgery detection techniques.

of visual content [67,68]. For example, Kurosawa et al. [69] utilized dark current noise as a unique fingerprint for device identification. However, since this noise can only be extracted from dark frames, the approach is limited to videos containing such frames. Lukas et al. [70] proposed using sensor pattern noise, particularly photo-response non-uniformity (PRNU), as an intrinsic fingerprint for source camera identification. Among existing methods, sensor pattern noise-based techniques have demonstrated the most reliable results. However, this active detection approach has limitations, such as possible visual quality degradation and limited device support. As a result, active detection techniques have seen less focus, despite aiming to verify the integrity of source data [54, 55].

In cases where digital media lacks embedded information such as watermarks or digital signatures, *passive detection* techniques are applied to assess authenticity. These methods do not rely on any pre-installed data but instead analyze the intrinsic characteristics of the content. For this reason, they are often referred to as passive or blind approaches. In passive detection, a commonly used strategy for video authentication is to analyze the continuity and consistency between frames based on visual quality, bitrate, and other inherent features. For images, spatial inconsistencies are examined to identify signs of tampering. This approach is considered more advanced and widely adopted in digital forensics due to its broad applicability and high effectiveness [56–58, 71].

Under passive detection strategy, double compression detection (see Figure 1.7) is a crucial forensic approach for identifying tampering in digital images and videos. Widely used formats like Joint Photographic Experts Group (JPEG) and Moving Picture Experts Group (MPEG) often lead to recompression when manipulations such as splicing, object insertion, or frame editing are applied and the content is saved again. This results in compression artifacts that differ based on the alignment and quantization settings of the original and modified versions. Although double compression

Table 2.1: Comparison between active and passive forgery detection approaches

Attribute	**Active Approach**	**Passive Approach**
Footprints	Requires prior knowledge such as digital watermarks or signatures to verify the authenticity of the video.	Does not require any prior embedded information; detection is based on intrinsic properties of the video.
Hardware Requirement	Involves additional hardware to embed information into each frame during recording.	Does not require specialized hardware, making the videos more susceptible to forgery.
Method Complexity	Generally involves simpler detection methods, as the process focuses on verifying embedded data.	Detection methods are more complex due to the absence of explicit clues, relying on content analysis.

is not a forgery itself, it serves as a reliable forensic trace associated with both intra frame and inter frame manipulation. Feng et al. [72] and Pevný et al. [73] proposed methods that extract features from image histograms and DCT coefficients, using support vector machines for detection. Lukáš and Fridrich [74] introduced the concept of double quantization artifacts, where periodic patterns in the DCT coefficient histograms—such as double peaks and missing centroids—indicate the presence of recompression and possible tampering. Similar attempts to detect forged video can be found in the works reported in Wang and Farid [75] demonstrated that double MPEG compression introduces larger motion estimation errors, which can be used to detect tampering. In a follow-up study [76], they further analyzed double compression effects by examining the distribution of DCT coefficients in I-frames across macro-blocks. They identified forgeries by measuring contrast deficiencies using the Euclidean distance. However, this method is less effective when frames are inserted or deleted to maintain the GOP structure. Sun et al. [77] extended the work of Chen and Shi [78] by proposing a method to determine whether the bitrate of a double-compressed video is higher than that of a single-compressed one. A key limitation of their approach is that both compression stages used the same MPEG-2 encoder, which reduces detection reliability when different encoders are applied in real-world scenarios.

As discussed in Chapter 1, section 1.3, image falsification can be broadly categorized into three main types: image splicing, image copy-move, and image retouching. In image splicing, regions from one or more images are copied and inserted into another image, creating a composite that appears authentic. Detection of such tampering often relies on inconsistencies in lighting and texture. For example, Hsu et al. [79] proposed a fully automated method for detecting image splicing by integrating three key components: geometry-invariant camera response function (CRF) estimation, consistency

verification, and image segmentation. Another approach, based on the Hilbert–Huang Transform (HHT), was introduced in [80], using Support Vector Machine (SVM) classifiers and reporting a detection accuracy of 80.15 percent. Additionally, Li et al. [81] enhanced this approach by combining moment-based and HHT-based features with SVM. Moreover, Shi et al. [82] proposed a natural image model that utilizes statistical features extracted through multi-size block DCT, including wavelet subband moments and Markov transition probabilities. Dong et al. [83] explored splicing detection using statistical features derived from run-length representations and edge statistics. In addition to image splicing, image copy-move forgery is another major concern due to its subtle nature and potential for social and legal misuse. In this form of forgery, a part of the image is copied and pasted within the same image to conceal or duplicate content. Numerous studies [84–93] have addressed this issue, highlighting its significance in real-world scenarios. Furthermore, image retouching represents another form of alteration where the intention may not always be malicious, but the original content is still modified. It involves techniques such as sharpening, blurring, adjusting brightness or contrast, and applying filters. Stamm et al. [94] proposed a detection method based on gray-level histograms to identify contrast enhancement by modeling the distribution of unaltered images. Similarly, Cao et al. [95] introduced a method to detect sharpening modifications by measuring gradient aberrations in the gray histograms of unsaturated luminance regions. Likewise, Ding et al. [96] proposed a technique that employs edge-perpendicular binary coding to detect texture modifications introduced by unsharp masking sharpening.

Alternatively, a range of specialized strategies is adopted to detect different forms of video forgery, as described in Chapter 1 and Section 1.4. Intra-frame forgeries where individual frames are manipulated similarly to images can often be detected using image manipulation detection techniques. In contrast, inter-frame forgeries such as frame insertion, deletion, duplication, or shuffling are typically exposed by identifying temporal inconsistencies in the video stream. A wide range of techniques has been proposed in the literature to address these challenges, utilizing both spatial and temporal features, motion analysis, and compression artifacts [97, 97–104]. Under spatio-temporal forgeries involve manipulations that affect both the spatial content of individual frames and the temporal consistency across frames. One of the most prominent examples of such forgery is deepfake, where the facial regions in video frames are synthetically altered or replaced using generative models, often leading to highly realistic yet deceptive content. These manipulations not only alter the visual features within frames but also attempt to preserve or mimic temporal continuity, making detection more challenging. Various deepfake detection methods have been proposed, leveraging spatial cues like

facial artifacts, blending inconsistencies, and biological signals, as well as temporal cues such as lip-sync mismatches and inconsistent facial dynamics [42, 105–110]. Such spatio-temporal analysis has become essential in developing robust detection systems against increasingly sophisticated deepfake techniques.

After analyzing various existing methods for digital image and video forgery detection under both active and passive approaches, this thesis primarily focuses on three significant areas: image copy-move detection, image/video deepfake detection, and frame duplication and deletion detection. These forms of forgery present serious concerns across social, legal, and security domains. Copy-move forgeries are often used to hide or duplicate parts of an image, commonly found in tampered news photos or manipulated legal evidence. Deepfake, which involve the manipulation of facial features to create highly realistic but fake images or videos of individuals, have been used to impersonate public figures, spread misinformation, and violate personal privacy—such as fabricating videos of politicians or celebrities. In the case of videos, frame duplication and deletion are typically employed to alter the sequence of events, particularly in surveillance footage or digital evidence. Due to the growing use and potential harm of these forgery techniques, this thesis concentrates on detecting such manipulations. The following sections present a detailed review of these key forgery types, including their mechanisms, detection approaches, and challenges.

Objectives of the Chapter:

The primary objectives addressed in this chapter are outlined as follows:

- To perform a comprehensive literature review on the selected image and video forgery detection techniques, examining both traditional and modern approaches used in the field.

- To identify the limitations and challenges associated with existing methods across different categories of forgery techniques, and to use these insights to guide the development of effective and robust detection strategies tailored to each type of forgery.

2.1 Image Copy-Move Forgery Detection Techniques

Copy-move forgery is a common image manipulation technique where a portion of an image is copied and pasted elsewhere in the same image to conceal or alter information [111]. Detection methods for copy-move forgery are generally fall into three major

approaches: (a) keypoint-based approach, (b) block-based approach, and (c) deep learning-based based that are described in details here.

2.1.1 Keypoint-based Approach

The methods fall under this category rely on feature extraction techniques such as Harris corner detector, Scale-Invariant Feature Transform (SIFT) and Speeded-Up Robust Features (SURF) to detect keypoints. Matched keypoints help detect duplicated regions, making keypoint-based methods highly robust to scaling, rotation, and viewpoint changes. Techniques like SIFT [112] are widely used for CMFD due to their effectiveness against various transformations like scale, rotation, and illumination changes [84, 113, 114]. Numerous researchers have proposed SIFT-based methods [115–118] to improve the detection process. While SIFT offers rotational and scaling invariance, it is limited to image flipping [119]. A limitation of SIFT-based feature extraction is the inadequate stability of image representation, leading to low discriminability and robustness. To overcome these challenges, recent studies have explored alternative descriptors, including KAZE [120], SURF [85], mirror-SIFT [88], and binarized SIFT [118]. Several previous studies [121–123] have focused on optimizing feature-level or detector-level integration. However, these techniques often impose a significant computational cost.

Emam et al. [86] utilized SIFT descriptors for feature extraction and employed the best-bin-first (BBF) search strategy to match similar features. Pan and Lyu [124] estimated the geometric transformation between matched SIFT keypoints to identify duplicated regions. Amerini et al. [16, 87, 125] introduced several SIFT-based CMFD approaches. Specifically, in [87], the authors applied maximum likelihood estimation (MLE) and the Random Sample Consensus (RANSAC) algorithm to estimate geometric transformations. Later, in [16], the researchers proposed a generalized keypoint matching technique, named 2-NN test for localizing multiple duplicated regions, using agglomerative hierarchical clustering to group potential cloned areas. Jin and Wan [126] improved the performance of SIFT-based CMFD techniques by integrating non-maximum value suppression and an optimized J-Linkage method. In [125], the J-Linkage algorithm was incorporated to enhance their previous methods. In another study, Anand et al. [127] extracted SIFT keypoints from actual image parts generated through dyadic wavelet transform (DyWT), rather than using grayscale images as preprocessing. Differently, Gong and Guo [128] used the color gradient of suspicious images as the sole input for SIFT extraction. Conversion to the Hue-Saturation-Value (HSV) color space was performed by Panzade et al. [129]. Zhao and colleagues [130, 131] proposed a SIFT-based CMFD method that leveraged particle

swarm optimization (PSO) to address parameter settings. Warif et al. [88] combined a SIFT-based CMFD framework with symmetry-based matching to enhance detection. Rajput et al. [132] used brute-force matching and DBSCAN clustering, with outlier detection to minimize false positives. However, it is sensitive to noise and compression artifacts. Deb et al. [133] combined SIFT, Accelerated-KAZE (AKAZE), and Histogram of Oriented Gradients (HOG) descriptors with Sum of Squared Differences (SSD) and Nearest Neighbor Distance Ratio (NNDR) matching, followed by RANSAC for false match removal. While more accurate than traditional methods, it needs fine-tuning for efficiency.

Shivakumar and Baboo [134] designed a CMFD approach based on SURF features, using a k-d tree for efficient feature matching. The authors of [135] applied SLIC segmentation before using SURF to locate duplicated regions. Mishra et al. [136] merged SURF with hierarchical agglomerative clustering (HAC) to develop a new CMFD method. Following multi-scale analysis and voting processes, Silva et al. [137] proposed a SURF-based CMFD scheme. Yang et al. [138] addressed the issue of sparse keypoints in uniform image areas by integrating adaptive minimal-maximal suppression (AMMS) with SURF. Sunitha et al. [139] combined SURF and SIFT descriptors with hierarchical clustering and mismatch elimination models. Although it enhances precision, it struggles with uniform textured images due to sparse keypoints. Alhaidery et al. [140] used SURF-HOG features, SLIC segmentation, and overlapping Zernike moments for robust detection.

Further, Chen et al. [141] suggested a CMFD method combining Harris corner detection with step sector statistics, applying the BBF algorithm for finding duplicated areas. Zhao and Zhao [142] fused Harris corners with LBP to detect region duplication. Wang et al. [143] employed statistical features from Harris keypoint neighborhoods as forensic features, enhancing matching accuracy with a novel feature matching method. Uliyan et al. [144] incorporated angular radial partitioning alongside Harris keypoints to propose a new CMFD technique.

2.1.2 Block-based Approach

This approach divides the image into overlapping or non-overlapping blocks and computes feature descriptors such as DCT, DWT, or Principal Component Analysis (PCA). Similar blocks are then matched to detect forgery. Fridrich et al. [145] started block-matching for CMFD, where overlapping image blocks undergo DCT to extract features. These features are sorted lexicographically, and similar blocks are marked as forged. To improve efficiency, Popescu and Farid [146] later applied PCA to reduce computations and processing time. Further advancements have since enhanced

detection speed and accuracy.

Babu and Rao [89] decomposed images using Steerable Pyramid Transform (SPT) and Gray Level Co-occurrence Matrix (GLCM) features from each orientation are used to train and classify with an Optimized SVM (OSVM). If forged, the image is divided into overlapping blocks, and forged regions are localized using block-wise GLCM features with similarity and distance thresholds. In another work, Babu and Rao [147] used Polar Complex Exponential Transform features, reduced dimensionality with Gradient Direction Pattern histograms, and removed false matches via windowing and morphological operators. Babu et al. [148] employed advanced feature descriptors such as Local Ternary Pattern, Local Phase Quantization, and others, with OSVM and Optimized Naïve Bayes Classifier (NBC) for forgery detection, even under attacks like compression, scaling, rotation, and brightness variations.

In 2018, Chen et al. [149] proposed a block-matching method tested on FAU and GRIP datasets. In continuation, Chen et al. [150] improved efficiency but faced delays in generating the localized map. In another work, Mahmood et al. [151] converted RGB to YCbCr and processing chrominance with Stationary Wavelet Transform (SWT) and DCT, verifying features through thresholding and morphological operators to remove false matches. In the work [152], the researchers employed a Doubly Stochastic Model (DSM) and Extreme Learning Machine (ELM) for classification, outperforming SVM but not OSVM.

Rathore [153] divided test images into overlapping blocks, using Bi-orthogonal Wavelet Transform (BWT) and Singular Value Decomposition (SVD) to extract feature vectors, comparing them using Minkowski distance and thresholds, with duplicate detection enhanced by Relevance Vector Machine (RVM). Kumar [154] proposed a reduced feature-based algorithm using Stationary Wavelet Transform (SWT), DCT, and SVD to extract features, reducing computational overhead with three key feature vectors for efficient forgery detection. Ganguly et al. [155] used Local Tetra Pattern (LTrP) for block-level comparison, dividing the image into overlapping blocks, extracting LTrP features, and matching similar blocks. False matches in uniform regions were removed using a shift vector-based outlier removal method.

2.1.3 Deep Learning-based Approach

CNNs are trained to recognize copy-move forged regions by analyzing spatial inconsistencies in images. In the early stages of the deep learning era, methods were developed to detect whether images were altered using copy-move forgery. Rao and Ni [156] proposed initializing the first layer of the CNN with a basic high-pass filter set instead of using a random strategy. This set, used for calculating residual maps in the spa-

tial rich model (SRM), acts as a regularizer, efficiently suppressing image content and capturing subtle tampering artifacts. Hosny et el. [90] introduced a precise CNNs architecture, which is computationally lightweight, featuring an optimal number of convolutional and max-pooling layers. A CNN-based CMFD is also proposed by Nikalje and Mane [157]. This method used patch sampling and modulus LBP to pre-train the neural network for feature learning and extraction. The extracted features are then fed into an SVM classifier to detect forged images. Elaskily et al. [158] used a CNN to learn hierarchical feature representations from input images, which are then utilized to distinguish between tampered and original images. However, these methods had limitations in precisely localizing and identifying the forged regions with sharp borders.

Advancements in image segmentation using deep learning have enabled methods to localize tampered regions and calculate accuracy by counting each tampered pixel in the duplicated area. Accordingly, Zhang et al. [91] employed a cross-layer intersection mechanism with a dense U-Net. They achieved this by using multi-stage training on the DenseNet model to store feature maps and enhance network convergence. An end-to-end neural network, AR-Net, suggested by [159] with adaptive attention and residual refinement. It fused position and channel attention for better feature representation, used deep matching for self-correlation, and employed atrous spatial pyramid pooling to create a coarse mask. Finally, the residual refinement module optimized the mask, preserving object boundaries. The approach of Liu et al. [92] was based on deep learning architecture, which consists of two stages: the first stage involves deep matching, while the second stage addresses incomplete areas and eliminates false alarms. Due to the employment of both deep matching and key point matching in the detection process, the method incurs computational costs. Sabeena and Abraham [160] utilized the Convolutional Block Attention Module (CBAM) for feature extraction, segmentation, and forgery localization. CBAM integrates spatial and channel attention for enhanced feature representation, while deep matching computes self-correlation. Atrous Spatial Pyramid Pooling (ASPP) fuses correlation maps, and bilinear upsampling resizes the output to match the original image. Zhong and Pun [93] used Dense-InceptionNet for extraction of multi-dimensional, multi-scale features followed by deep feature correlations to generate three candidate matching maps. Later, these maps are combined using cross-entropies for enhanced training. Islam et al. [161] suggested a dual-order attention model in which the first-order attention is intended to gather copy-move location information, while second-order attention uses more discriminative characteristics for the patch co-occurrence. However, if the copied section is taken from the consistent background and repositioned onto that same background, it might fail to

detect. According to the investigation of Weng et al. [162], they used a U-Net structure model, where the primary concentration was in multiple cross-layer connections containing self-correlation calculation.

Identifying and differentiating the source and tampered regions is important in real-world CMFD. To address this issue, Wu et al. [163] introduced the initial CNN-driven framework for CMFD, known as BusterNet. This framework incorporates a two-branch architecture (manipulation and similarity branches) to classify and pinpoint the source and tampered regions. Chen et al. [164] presented an enhanced iteration of BusterNet by designing two consecutive subnetworks: the copy-move similarity detection network (CMSDNet) and the source/target discrimination network. The method fails to enhance network performance in terms of semantic segmentation. In CAMU-Net designed by Zhao et al. [165] hierarchical feature extraction to obtain multi-scale key feature maps followed by a feature matching stage that predicts copy-move forgery areas of varying scales. To enhance results, a coordinate attention-based resource allocation stage prioritizes relevant areas. In the up-sampling stage, they fuse high and low-level information to improve CMFD performance. Niloy et al. [166] exploited feature distribution differences to localize image forgeries. Using contrastive loss, they map features into a space where actual and manipulated regions are distinctly separated.

Limitations of Existing Image Copy-move Forgery Detection Techniques

After a critical analysis of various methodologies, several limitations in existing strategies were identified. The following points highlight key challenges observed in different approaches:

- Traditional keypoint and block-based methods often fail to detect small or texture-similar forgeries due to weak feature matching, leading to false positives or negatives.
- Block-based approaches often require high computational resources due to the processing of numerous image blocks or keypoints.
- Existing techniques underperform in flat or textureless regions, while block-based methods, though better in such areas, are computationally expensive and less effective under geometric distortions.
- Many existing approaches are not robust against post-processing operations such as noise addition or smoothing, which can obscure forgery traces.

- Detecting small duplicated regions remains a challenge, as some methods fail to localize fine-grained forgeries accurately.

- The effectiveness of copy-move detection methods depends heavily on the chosen feature descriptors, which may not perform consistently across different image types and forgery patterns.

2.2 Deepfake Detection Techniques

Deepfake forgeries can occur in both images and videos. This is commonly treated as a binary classification task, where models distinguish between real and fake videos. This approach requires a large dataset of authentic and fake videos for training. Research has primarily focused on deep learning techniques, categorized into CNN-based and RCNN-based methods. CNN-based methods extract one or more facial images from video frames and use a CNN for classification. To decide whether the video is real or manipulated by deepfake techniques, the final judgment is made either by majority voting over the extracted facial images or based on the analysis of a single facial image. However, these methods rely solely on spatial features from individual frames, without capturing temporal inconsistencies in deepfake videos. Video detection necessitates advanced techniques such as 3D CNNs or recurrent neural networks. The process involves analyzing multiple frames, often in real-time, making it computationally demanding [105, 167]. Efficient algorithms and high-performance computational resources are required to process information across an entire video swiftly [168].

The advent of powerful GAN models [169, 170] has made deepfake detection increasingly challenging. Earlier face manipulation detection methods primarily relied on hand-crafted features, utilizing image properties such as Discrete Fourier Transform (DFT) [171], facial landmarks [106], and camera-captured noise [172] to identify forged artifacts. These approaches aimed to reveal manipulation traces left by deepfake generation techniques. However, advancements in GAN models and sophisticated multimedia compression techniques have made these artifacts nearly imperceptible. Several researchers [42, 107, 173–175] have primarily relied on single CNN-based models to learn artifacts present in deepfake videos and images. Some studies [42, 107, 173] trained models from scratch, without using pre-trained weights, for image manipulation detection. For instance, Afchar et al. [42] introduced two CNN architectures: (1) MesoNet-4, consisting of four convolutional blocks followed by a fully connected layer, and (2) MesoInception-4, inspired by the Inception module [176], where the first two convolutional blocks of MesoNet-4 were replaced with those from the Inception model. Guo et al. [107] focused on detecting subtle manipulation traces by emphasiz-

ing manipulated content while suppressing image content. Their approach generates an intermediate feature map using convolutional layers, subtracts the original image from this map, and filters out irrelevant parts, allowing the network to concentrate on critical details.

Beyond common challenges in deep learning models, several studies [44, 108] have explored integrating attention mechanisms into CNNs to enhance focus on facial image regions. For instance, Nguyen et al. [177] introduced LAA-Net, a multi-task learning framework incorporating heatmap-based and self-consistency attention to highlight artifact-prone areas. Additionally, an Enhanced Feature Pyramid Network (E-FPN) efficiently propagates low-level features while minimizing redundancy in deepfake detection. Xia et al. [109] proposed MMNet, a Multi-Collaboration and Multi-Supervision Network, designed to handle diverse spatial scales and sequential variations in manipulated face images, enabling recovery without prior knowledge of the manipulation technique. However, these attention models primarily apply convolution operations across the entire image, potentially emphasizing irrelevant regions. The presence of non-essential features in existing models may mislead classifiers, increasing training time. Implementing an effective feature selection method could help mitigate this issue by filtering out unimportant features.

Existing models perform well when manipulation artifacts are known but struggle with unknown artifacts from different models. To address this, Wang et al. [110] analyzed the mouth region, combining global (entire face) and local (mouth region) features to improve generalization. Transformers have shown promise in enhancing deepfake detection. Wang et al. [178] leveraged transformer models to capture both local and global image features, while Heo et al. [179] employed an efficient vision transformer for extracting multi-scale features. Additionally, Wang et al. [180] developed a two-branch structural network integrating a cascaded multi-self-attention mechanism alongside EfficientNet-B4. Reis et al. [181] proposed a threshold classifier using similarity scores from a deep CNN trained for facial recognition, classifying videos as authentic or fake based on the highest similarity score. To address semantic inconsistencies and overfitting, Tian et al. [182] introduced a channel attention module that combines local and global contexts.

Previous methods mainly extracted spatial features from individual video frames, ignoring temporal dependencies and missing motion-related artifacts. Deepfake video detection now leverages temporal information using techniques like 3D CNNs and recurrent neural networks to capture frame coherence and subtle artifacts across time. Liu et al. [183] propose a lightweight 3D CNN with exceptional capability in integrating spatial information across the temporal dimension. The model incorporates a channel

transformation (CT) module to reduce parameters while enhancing feature extraction. Experimental results show that the proposed network surpasses previous deepfake detection methods. In another work, Zhang et al. [184] propose a Temporal Dropout 3D CNN (TD3DCNN) to detect deepfake videos by leveraging inconsistencies among frames. The model samples fixed-length frame volumes and processes them using a 3D CNN to extract multi-scale features. A temporal dropout operation randomly selects frames in each batch to enhance robustness. Building on previous methods that leverage 3D CNNs for deepfake detection, Lu et al. [34] enhances feature extraction by incorporating a 3D-attentional inception network with channel and spatial-temporal attention modules. Previous methods focused only on temporal information, while Lin et al. [185] leveraged both spatial and temporal inconsistencies. To enhance feature extraction, CBAM is integrated into the I3D network, and three feature extraction modules (RGB, optical flow, and noise) are introduced for deeper forgery detection. Almestekawy et al. [186] introduced a technique that combines spatiotemporal textures with deep learning-based features. Their approach utilizes an enhanced 3D CNN with a spatiotemporal attention layer within a Siamese architecture.

Limitations of Existing Deepfake Detection Methods

After a thorough analysis of existing deepfake detection techniques, several limitations have been identified. The following points summarize the key challenges faced by various approaches:

- Existing methods use CNNs that focus on restricted facial areas like the eyes or mouth, missing broader facial context and subtle artifacts.
- Models without attention mechanisms fail to capture wider contextual dependencies, limiting effective localization of manipulated regions.
- Approaches do not effectively combine detailed local features with global facial information, which can result in incorrect classifications.
- Deep models often generate numerous redundant features. Without effective feature selection, this increases complexity and slower processing.
- Methods rely only on deep features, overlooking the benefits of combining them with handcrafted features for capturing subtle artifacts and domain-specific cues that may enhance forgery detection performance.
- Few approaches apply one-time feature selection to retain only the most relevant features, while hierarchical feature reduction further shortens feature length and

can improve efficiency. However, such strategies are rarely explored for deepfake detection.

2.3 Inter-frame Video Forgery Detection Techniques

In digital forensics, video tampering detection remains in its early stages, facing a lack of robust techniques for accurately detecting and localizing tampered content [187, 188]. When frames are inserted or duplicated within a video, they disrupt the consistency of object motion at the boundaries of the manipulated segment. Various features have been explored in the literature to detect these inconsistencies, including texture features [57, 97, 103, 189], optical flow [102, 190, 191], prediction residuals [192–194], standard deviation of residual frames [195], the bag-of-words (BoW) model [196], correlation [97, 98, 104], motion residuals [197], and noise residue-based features [198].

Li et al. [99] investigated discontinuities caused by inter-frame tampering by applying a k-means clustering algorithm on the consistency pattern of 2D phase congruency to detect forgeries. While this method is effective for detecting and localizing frame insertion attacks, it struggles with identifying smaller frame deletions and trimming operations. Similarly, Zhang et al. [97] analyzed the Quotients of Correlation Coefficients of Local Binary Pattern (QoCCLBP) to detect frame insertion and deletion attacks. Another approach by Li et al. [199] leveraged inconsistencies in the Quotients of Mean Structural Similarity Index Measure (QoMSSIM), proving effective in detecting insertions and deletions even when an entire GOP is removed and demonstrating robustness against re-compression and white Gaussian noise. However, it still falls short in identifying small-scale frame manipulations.

In another work, Jin et al. [100] examined artifacts in content-oriented feature descriptors such as BRISK, SURF, and MSER but could only detect frame duplication attacks. Likewise, Singh et al. [200] proposed a technique based on inconsistencies in the mean feature, but it too was limited to detecting frame duplications. Another method [101] focused on MPEG streams, detecting tampering by analyzing variations in coding parameters and periodic artifacts in the DCT coefficients' high-frequency components. Despite its effectiveness, the approach lacks adaptability across different video codecs (e.g., AVC, HEVC, VVC) and struggles when tampering does not alter the coding structure, especially in complex GOP configurations. Kingra et al. [102] presented a method that analyzed temporal correlation inconsistencies between successive frames by studying variations in optical flow brightness gradients and prediction residuals. This method can detect frame insertion, deletion, and duplication—even

under large motion—but its performance degrades under high illumination conditions.

Bakas et al. [101] introduced a technique that uses prediction footprint variation (PFV) and motion vector variation (VMV) to detect and localize forgery in videos with varying compression levels. While GOP size-independent, it cannot detect insertion or deletion of an entire GOP. Zhao et al. [103] proposed computing histogram similarities (H–S and S–V) of adjacent frames, incorporating SURF features to detect various inter-frame attacks. However, this method relies on an improved shot boundary detection step and is ineffective in handling dynamically obtained shots, abrupt changes, or complex forgeries. Fadl et al. [201] explored differential energy function in frame residues to detect insertions and deletions, but the method fails with static scenes and cannot identify frame shuffling. Lastly, Shelke et al. [104] investigated correlation inconsistencies in adjacent frame patterns using entropy-based texture features like DistrEn2D and MSE2D. However, this method lacks effectiveness in detecting and localizing more complex types of forgeries.

Limitations of Existing Inter-frame Forgery Detection Methods

Inter-frame forgery detection methods play a crucial role in identifying video manipulations, such as frame duplication and deletion. However, existing approaches suffer from several drawbacks that limit their effectiveness in real-world scenarios. Addressing these limitations requires the development of more robust, context-aware, and computationally efficient inter-frame forgery detection methods. The key limitations of these methods are outlined below:

- Most inter-frame detection methods use pixel-wise comparisons, making them ineffective against subtle, modified forgeries.
- Many methods focus only on low-level features, overlooking temporal inconsistencies caused by frame duplication or deletion.
- Existing techniques are often tested on limited, non-standard datasets, reducing realism in evaluation.
- High sensitivity to noise, blur, and contrast changes limits performance on real-world, re-encoded videos.
- Existing deep learning-based methods depend on seen data and struggle to detect unseen forgeries.

2.4 Discussion

This chapter provided a comprehensive overview of image and video forgery detection strategies. Based on the most pressing challenges and current concerns in society and social platforms, it focused on reviewing three major types of digital media forgery: copy-move in images, deepfake in both images and videos, and inter-frame forgeries such as frame duplication and deletion. All techniques were studied under the passive detection approach, which analyzes content without relying on pre-embedded information making it more applicable to real-world use cases. The survey highlighted a shift from traditional handcrafted methods to deep learning and hybrid approaches. While deep models enhance detection accuracy, they often face limitations due to the restricted receptive field of CNNs, which can hinder their ability to capture global contextual information crucial for detecting subtle forgeries. This limitation in existing methods suggests that hybrid approaches combining handcrafted and deep along with the integration of attention mechanisms to expand the receptive field, may offer improved adaptability across deepfake forgery scenarios. Each forgery type presents unique challenges in CMFD demands fine localization, deepfake require semantic understanding, and inter-frame forgery involves detecting temporal inconsistencies.

Chapter 3

Copy-Move Forgery Detection in Images

In an era dominated by digital imagery, the manipulation of visual content commonly known as image tampering has become an increasingly widespread and pressing issue [202]. When maliciously altered images are used to fabricate social controversies, falsify criminal evidence, or manipulate academic achievements, they pose serious threats to societal stability and public safety. As a result, research in image tampering forensics holds significant importance.

One of the most common forms of image tampering is copy-move forgery, where one or more regions of an image are copied and pasted within the same image (see Figure 3.1). During this process, the duplicated regions may undergo transformations such as rotation or scaling. This type of forgery alters the original semantic content of the image by replicating a part of it and relocating it to another area within the same image, as visually illustrated in Figure 3.1. In this context, the duplicated region is referred to as the source area, while the location it is pasted to is called the tampered area; collectively, these are known as target areas. The primary objective of CMFD is not merely to determine whether an image has been tampered with (i.e., a binary classification task), but rather to localize the forged regions within the image. This task is particularly challenging because the copied and pasted regions originate from the same image, resulting in nearly identical photometric properties that make detection more difficult. Over the years, CMFD has been widely studied, leading to the development of numerous detection techniques. These approaches include handcrafted feature-based methods, such as block-based and keypoint-based techniques [89, 122, 139, 140, 149, 153, 203], as well as deep learning-based methods [163, 204–207], each contributing unique advantages to the field.

Objectives of the Chapter

In sort the objectives that have been covered in this chapter are as follows:

- Design lightweight model for real-time image CMFD while maintaining high accuracy.
- Improve pixel-level forgery detection using deep learning to precisely identify manipulated regions.

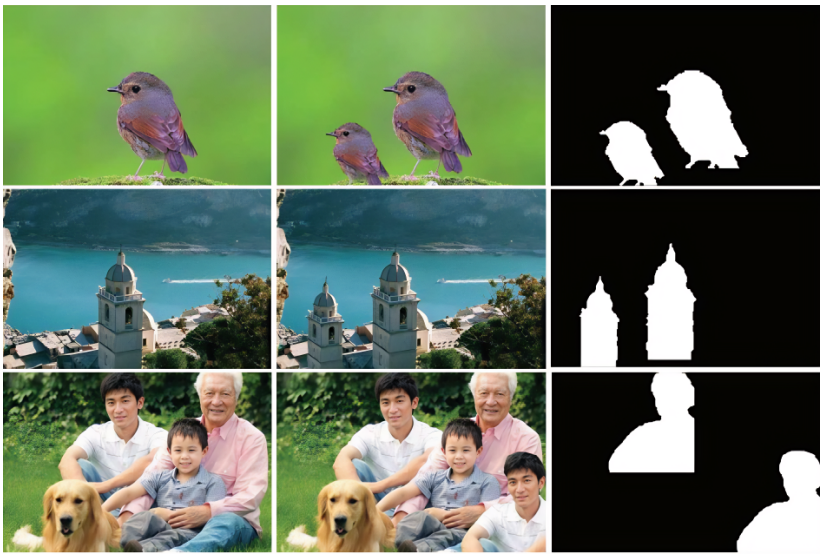


Figure 3.1: Examples of copy-move forgery images are shown, where the first column displays the original images, the second column shows the forged images, and the third column presents the corresponding ground truth masks [204].

- To extend the detection capabilities of the proposed model by ensuring robustness against various post-processing operations such as compression, brightness adjustment, color reduction, image blurring, noise addition, and contrast adjustment.

- Train models on diverse datasets to ensure reliable detection across different image formats and resolutions.

To address the defined objectives, a deep learning-based approach is adopted to address the identified limitations. Unlike traditional keypoint and block-based methods, deep learning models can automatically learn and extract discriminative features without manual design, making them more effective in detecting small or texture-similar forgeries. Traditional methods, such as keypoint-based and block-matching techniques, often struggle with handling post-processing operations like scaling, rotation, and compression. Deep learning, particularly CNNs and attention mechanisms, has demonstrated significant improvements in feature extraction and forgery localization. In this chapter, the proposed detection method is presented, leveraging deep neural networks to enhance robustness, accuracy, and computational efficiency.

3.1 Methodology

This research enhances the MultiResUNet architecture [41] for CMFD by leveraging its multi-scale feature extraction capability. To optimize efficiency, the standard convolutional layer in the MultiRes Block (MRB) is replaced with a separable con-

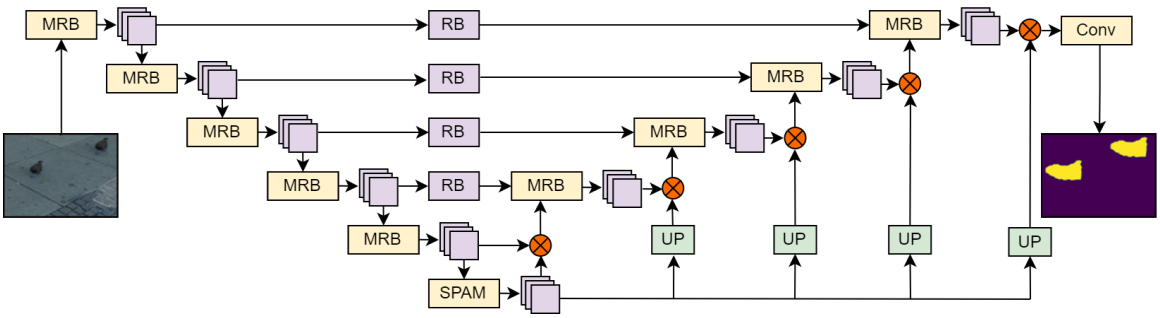


Figure 3.2: The block diagram illustrates the proposed modified MultiResUNet architecture incorporating the SPAM attention module. MultiRes blocks (MRB) are employed to extract meaningful features from the input image. Residual connections (RB) between encoder and decoder features ensure efficient information flow across multiple network levels. The attention-enhanced bottleneck feature map undergoes upsampling (UP) and is subsequently multiplied with the decoder feature map to refine model performance. Finally, a convolutional layer with sigmoid activation (Conv) generates the predicted binary segmentation mask.

volution, reducing model parameters for real-time applications. Residual connections mitigate vanishing gradients, facilitating seamless information flow between encoder and decoder layers, leading to a more effective and efficient model. The model introduces the Similarity-based Positional Attention Module (SPAM) to enhance CMFD by leveraging patch-wise similarity estimation. Operating on the bottleneck feature map, it identifies regions with high cosine similarity, refining detection. Additionally, the Positional Attention Module (PAM) highlights spatially similar areas, enhancing the feature map. The attention enhanced features are then integrated with decoder features at multiple levels for segmentation. Finally, a sigmoid attention function in the last convolutional layer generates the segmentation mask. The combination of SPAM and the lightweight modified MRB improves CMFD performance while maintaining efficiency for real-time applications. A visual overview is shown in Figure 3.2.

3.1.1 MultiResUNet

The MultiResUNet is an advanced framework designed for image segmentation and medical image analysis, enhancing the U-Net architecture with MultiRes Blocks (MRBs) and Residual Blocks (RBs). The core of MultiResUNet lies in its MRB, which captures multi-scale information by integrating features from multiple pathways. Similar regions may appear close together or far apart, varying in size, underscoring the need for multi-scale feature extraction to capture both global and local information. RBs serve as essential components that facilitate gradient flow and information exchange across layers, aiding in the training of deeper networks while mitigating gradient-related issues [208]. To enhance efficiency and enable real-time processing,

both MRBs and RBs utilizes separable convolutional layers instead of traditional convolutions. This modification significantly reduces the model's trainable parameter while maintaining performance, as demonstrated in Table 3.1. The ability of MRBs to extract multi-scale features makes them highly effective for CMFD and segmentation tasks. With their multiple pathways, they excel in complex image analysis, accurately detecting and segmenting manipulated regions.

3.1.2 SPAM

Building on traditional ML techniques that primarily use region similarity-based CMFD, a patch-wise similarity-based attention network is introduced to detect forged regions. This approach utilizes the bottleneck layer feature map, B, to extract patches for analysis. Initially, B with dimensions $H \times W \times C$ undergoes a separable convolutional transformation with a single filter, reducing it to $H \times W \times 1$. This step significantly lowers computational complexity. As illustrated in Figure 3.3, cosine similarity is computed between two identical copies of B (denoted as B_1 and B_2). The similarity between corresponding patches (e.g., patch a in B_1 and patch a in B_2) is inherently high, as they are identical ($B_1 = B_2 = B$). Consequently, the cosine similarity for patches aa, bb, cc, and dd is initialized to 0. For all other comparisons, patches with similar content indicating forged regions produce higher similarity values, while dissimilar patches yield lower values. Finally, patch values such as a' are computed using Equation 3.1, where the 4 in the denominator represents the total number of patches, with similar calculations applied to generate b', c', and d'.

$$\begin{aligned} a' &= \frac{aa + ab + ac + ad}{4}, \quad b' = \frac{aa + ab + ac + ad}{4} \\ c' &= \frac{aa + ab + ac + ad}{4}, \quad d' = \frac{aa + ab + ac + ad}{4} \end{aligned} \tag{3.1}$$

The cosine similarity scores (a', b', c', and d'), being scalar values, are multiplied by their corresponding patches to generate a similarity attention-aided feature map B_{out} (say, $A = a' \times a$ in Figure 3.3). This patch-wise similarity-based attention network leverages the inherent similarity characteristics of copy-move forgery, enhancing the model's ability to detect forged regions more effectively. On B_{out}, the Position Attention Module (PAM) is applied, enhancing local features by incorporating a broader contextual understanding, thereby improving their representational capacity. The process starts with a feature map $B_{out} \in \mathbb{R}^{C \times H \times W}$. A convolutional layer transforms this feature map into two new feature maps, X and Y, both having dimensions $\mathbb{R}^{C \times H \times W}$. Next, X and Y are reshaped into matrices of dimensions $\mathbb{R}^{C \times N}$, where $N = H \times W$,

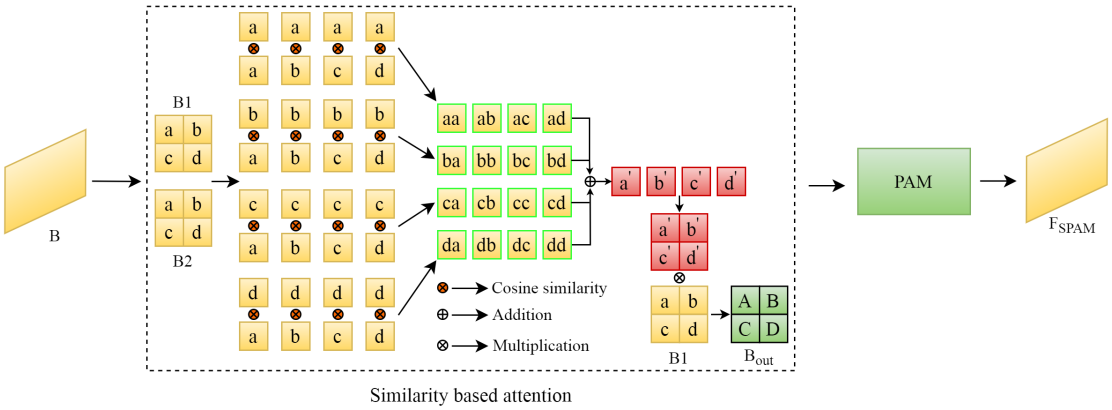


Figure 3.3: An illustration of the SPAM attention module.

representing the total number of pixels in the feature map. This step is followed by a matrix multiplication between the transpose of Y and X followed by a softmax layer application, expressed as:

$$S = \text{softmax}(Y^T \cdot X), \quad S \in \mathbb{R}^{N \times N} \tag{3.2}$$

This spatial attention map S captures interdependencies among different pixel locations in the feature map. PAM utilizes this attention mechanism to highlight significant spatial information in local features. To refine the representation, B_{out} is used to generate two feature maps, F_a and F_b, through convolution layers with a kernel size of 1. Each of these feature maps has a dimension of $H \times W \times C$. The feature map F_a is further processed to derive F_{sim}, which captures feature covariance across spatial dimensions. Global Attention Pooling (GAP) is then applied to F_{sim}, transforming it into F'_{sim}. The refined feature representation is obtained by multiplying F'_{sim} with B_{out}, followed by a sigmoid activation:

$$F_{eattn} = sigmoid(F'_{sim} \cdot B_{out}) \tag{3.3}$$

Finally, the enhanced attention-based feature representation is obtained by multiplying F_b with F_{eattn}, yielding the final output:

$$F_{SPAM} = F_b \cdot F_{eattn} \tag{3.4}$$

This approach improves the ability of local features to capture complex patterns and structures in the input data. The detailed architecture of PAM is illustrated in Figure 3.4. The mathematical formulation is provided in Equation 3.5, where s_{ji} quantifies the influence of the ith position on the jth position.

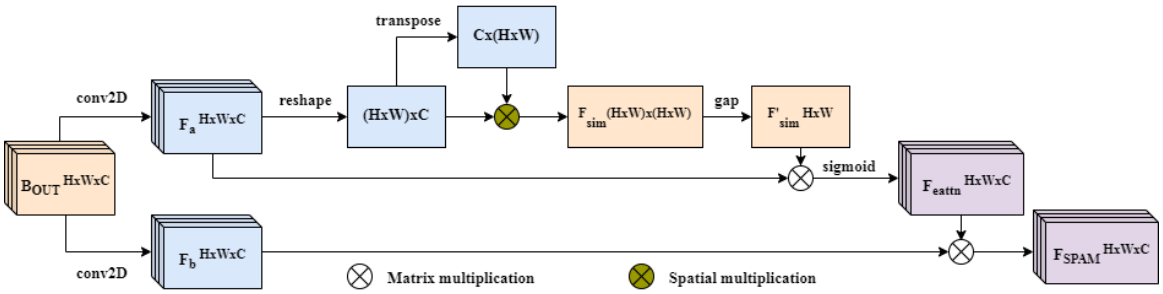


Figure 3.4: Positional Attention Module (PAM).

$$s_{ji} = \frac{\exp(X_i \cdot Y_j)}{\sum_{i=1}^{N} \exp(X_i \cdot Y_j)} \tag{3.5}$$

Since the computation involves two nested loops iterating over N_{patch} elements and performing operations such as multiplications, additions, and cosine similarity calculations, the overall time complexity can be approximated as $O(N_{patch}^2 \times K)$, where K represents the computational complexity of the operations within the loop body per iteration. The space complexity is primarily determined by the input feature set B, along with intermediate representations B_1, B_2, and the output B_{out}. Given that each feature map maintains a similar size, the overall space complexity remains approximately linear with respect to the input size B, scaling proportionally to the number of features.

Following this, the feature map F undergoes a convolutional transformation, producing a new feature map $D \in \mathbb{R}^{C \times H \times W}$. This transformed feature map is then reshaped into $\mathbb{R}^{C \times N}$, after which a matrix multiplication is performed between D and the transpose of S, resulting in an output of dimensions $\mathbb{R}^{N \times C}$. The result is subsequently reshaped back to $\mathbb{R}^{C \times H \times W}$. Finally, the feature map is scaled by a learnable parameter α and added element-wise to B_{out}, producing the final output F_{SPAM}, as formulated in Equation 3.6.

$$F_{SPAM_j} = \alpha \sum_{i=1}^{N} (s_{ji} \cdot D_i) + B_{out_j} \tag{3.6}$$

The parameter α is initialized to 0, making it learnable and enabling adaptive weight assignment during training. Consequently, the resulting feature F_{SPAM} at each spatial location represents a weighted aggregation of contextual features across all positions, combined with the original features. This mechanism facilitates a global contextual perspective while selectively integrating relevant spatial information through the attention map, thereby ensuring semantic consistency in feature representations.

Ultimately, F_{SPAM} serves as the attention-enhanced feature map obtained by applying the SPAM mechanism on B.

3.1.3 Evaluation Metrics

To evaluate the performance of the proposed image CMFD methods, four commonly used evaluation metrics are employed: Accuracy, Precision, Recall, and F1-Score. These metrics are computed based on pixel-level classification results, where each pixel is categorized as either part of a forged (tampered) region or a genuine (untampered) region. Let TP represent the number of correctly identified tampered pixels (true positives), TN represent the correctly identified untampered pixels (true negatives), FP denote the untampered pixels incorrectly labeled as tampered (false positives), and FN denote the tampered pixels incorrectly labeled as untampered (false negatives).

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{3.7}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{3.8}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{3.9}$$

$$\text{F1-Score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{3.10}$$

These metrics enable an objective evaluation of the model's capability to accurately localize tampered regions at the pixel level, which is crucial in CMFD tasks.

3.1.4 Loss Function

For CMFD tasks, model performance is evaluated using two essential loss functions: Intersection over Union (IoU) loss and Binary Cross Entropy (BCE) loss. These losses are crucial for predicting binary segmentation masks that identify forged regions. The IoU loss quantifies the overlap between the predicted and ground truth masks. It is computed using Equation 3.11, considering TP, FP, and FN at the pixel level.

$$\text{IoU Loss} = 1 - \frac{TP}{TP + FP + FN} \tag{3.11}$$

Conversely, the BCE loss evaluates the discrepancy between predicted and actual segmentation masks. As defined in Equation 3.12, it incorporates N, the total number of pixels, y_i, the ground truth label of pixel i, and p_i, the predicted probability for the foreground class.

$$\text{BCE Loss} = -\frac{1}{N}\sum_{i=1}^{N}\left(y_i \log(p_i) + (1 - y_i)\log(1 - p_i)\right) \tag{3.12}$$

To enhance model performance, the proposed approach employs a composite loss function that integrates both IoU loss and BCE loss, as expressed in Equation 3.13.

$$L_{seg} = \text{IoU Loss} + \text{BCE Loss} \tag{3.13}$$

This combined loss formulation ensures effective optimization, improving the accuracy and robustness of the model in detecting copy-move regions.

3.1.5 Experimental Setup

The proposed model is implemented in Python, utilizing TensorFlow and Keras for efficient deployment. Libraries such as numpy, OpenCV, and scikit-learn are employed for data preprocessing and manipulation. Training is accelerated using an NVIDIA TESLA P100 GPU to maximize computational efficiency. All input images are resized to 256×256 pixels for standardization. A batch size of 8 is chosen to optimize resource usage and convergence. The Adam optimizer, with a learning rate of 0.001, is used to dynamically adjust weights, improving training stability and efficiency.

3.1.6 Dataset Description

The model has been evaluated using four datasets: CoMoFoD [209] and COVERAGE [210], CASIA TIDE v2.0 [211] and MICC-F600 [125]. The CASIA TIDE v2.0 dataset is a benchmark for image forgery detection, containing 7,491 authentic and 5,123 manipulated images, including approximately 3,274 copy-move and 1,849 splicing forgeries. A dedicated subset, CASIA-CMFD, focuses on CMFD and includes 1,313 forged images with their 1,313 authentic counterparts, totaling 2,626 samples. It is notable for its manually created manipulations and utility in evaluating passive detection methods.

The CoMoFoD dataset contains 5,000 forged images, derived from 200 base images without post-processing and 4,800 tampered images subjected to various post-processing techniques. These techniques include JPEG compression (JC), brightness adjustment (BC), color reduction (CR), image blurring (IB), noise addition (NA), and contrast adjustment (CA), each applied at different intensity levels. This process produced 4,800 post-processed forged images, enabling the assessment of the model's robustness against different attack scenarios. The images are named in the format N1_M1, where N1 is a three-digit image number, and M1 represents one of four marks: "F" for forged images, "B" for binary masks (black and white), "M" for colored masks,

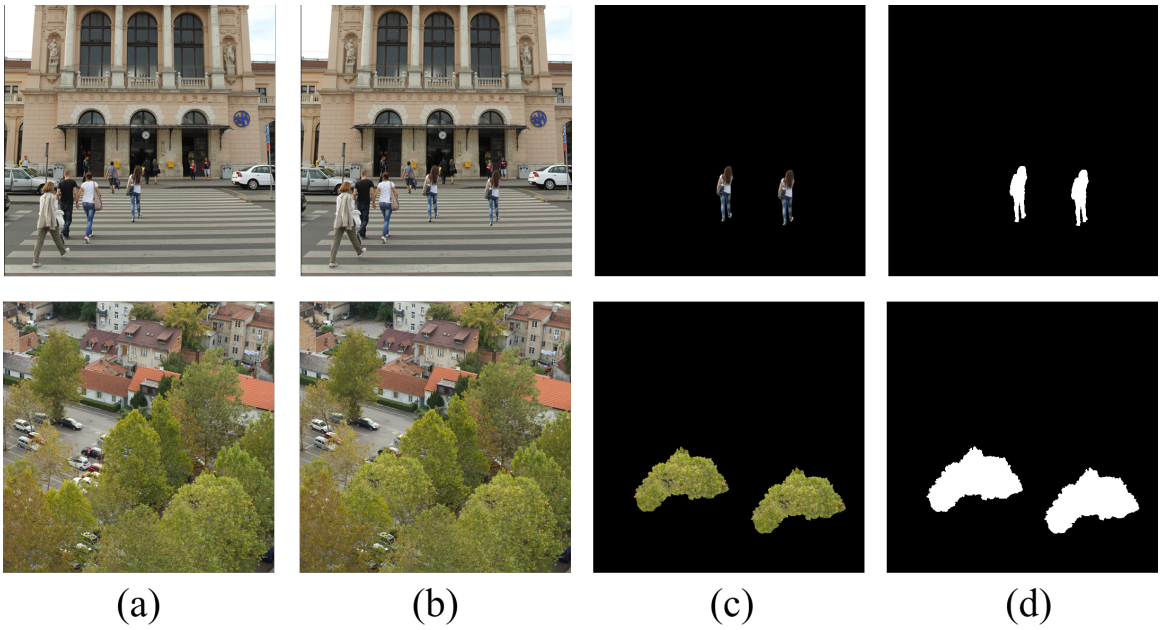


Figure 3.5: Sample images from the CoMoFoD dataset: (a) Real image, (b) Forged image with copy-move manipulation (c) Corresponding mask highlighting the manipulated regions, and (d) Binary mask indicating the exact forged areas.

and "O" for original images.

The COVERAGE dataset consists of 100 forged images, each accompanied by a ground truth mask, with image resolutions ranging from 190 × 334 to 551 × 556 pixels. The MICC-F600 dataset consists of 160 manipulated images and 440 authentic images, with resolutions ranging from 800 × 533 to 3888 × 2592. For all experiments, a 7:2:1 training-validation-testing ratio is applied across all datasets. Figure 3.5 shows some sample real and forged images from CoMoFoD dataset with corresponding mask image.

3.1.7 Ablation Study

An ablation study is conducted on the CoMoFoD dataset to determine the optimal configuration and parameters for the proposed model. The analysis focuses on evaluating the overall network architecture. The patch size is selected based on the dimensions of the bottleneck feature map (B), which measures $16 \times 16 \times 512$. The experimental setups are outlined as follows:

(i) MultiResUnet
(ii) Modified MultiResUnet (convolutional layers replaced by separable convolutional layer)
(iii) Modified MultiResUnet + SPAM (patch size = 2)
(iv) Modified MultiResUnet + SPAM (patch size = 4)
(v) Modified MultiResUnet + SPAM (patch size = 8)

Table 3.1 demonstrates the significant impact of the SPAM attention module in improving performance. A considerable reduction in parameters (from 7.27 million to 2.76 million) is observed by replacing standard convolutional layers with separable convolutions, without notably affecting model accuracy. Since the bottleneck layer encodes semantically rich and compact features, SPAM is applied to it. The results indicate that a patch size of 4 yields the best performance. This is because the feature map B has dimensions $16 \times 16 \times 512$, making a patch size of 8 too large, leading to a broader region being highlighted and increasing false positives. Since precision is inversely related to false positives, it is lower for a patch size of 8 compared to 4, as shown in Table 3.1. On the other hand, a patch size of 2 is too small, leading to excessive fragmentation of the feature space, which hampers the attention module's ability to capture meaningful regional patterns, which increases the likelihood of false positives or false negatives.

Table 3.1: CMFD results for different model architectural configurations based on standard metrics. All values, except for the number of parameters, are presented in %.

Model	**Parameters (in million)**	**F1-score**	**Accuracy**	**Precision**	**Recall**
(i)	7.27	74.30	98.48	78.06	72.77
(ii)	2.76	73.46	98.41	78.09	71.85
(iii)	3.26	84.08	99.18	89.99	80.86
(iv)	3.26	84.51	99.34	92.30	81.92
(v)	3.26	80.55	99.07	91.25	76.25

With a patch size of 4 (as it yields the best results according to Table 3.1), the impact of combining PAM and Similarity-based attention is evident in Table 3.2. PAM enhances recall, while Similarity-based attention improves precision. Integrating both techniques within SPAM provides a notable improvement in both precision and recall, further enhancing the F1-score.

Table 3.2: Performance evaluation of SPAM components using standard metrics. All values are presented in %.

Model	**F1-score**	**Accuracy**	**Precision**	**Recall**
PAM	75.39	98.62	74.98	79.14
Similarity-based attention	79.36	99.02	86.83	74.75
SPAM	84.51	99.34	92.30	81.92

To further analyze the impact of different distance metrics in SPAM (with a fixed patch size of 4), Table 3.3 highlights the effectiveness of combining PAM and Similarity-based attention. Cosine similarity is less sensitive to outliers as it measures the angle

between vectors, whereas Manhattan and Euclidean distances rely on absolute differences, making them more susceptible to outliers. Additionally, the table shows that Pearson's correlation is significantly influenced by noise and non-linear feature relationships, which are better managed by cosine similarity.

Table 3.3: CMFD results for different distance metrics used in SPAM based on standard evaluation metrics. All values are expressed in %.

Distance	**F1-score**	**Accuracy**	**Precision**	**Recall**
Euclidean	81.32	99.10	90.83	76.04
Manhattan	77.15	98.90	86.90	72.96
Pearson's correlation	73.90	98.89	90.86	66.97
Cosine similarity	84.51	99.34	92.30	81.92

3.1.8 Comparison with Past Methods

From Figure 3.6, it is clear that the model demonstrates strong performance across all datasets. The proposed model effectively detects forged regions, even when they are arbitrarily rotated, of smaller dimensions, or difficult to spot within natural scene images. A comparative analysis with SOTA algorithms is presented in Table 3.4, Table 3.5, Table 3.6, and Table 3.7, corresponding to the CASIA v2, CoMoFoD, COVERAGE, and MICC-F600 datasets, respectively. While the proposed model lags behind UCM-Net in recall on the COVERAGE dataset, its higher precision improves the overall performance, resulting in a superior F1-score compared to UCM-Net.

Table 3.4: Comparison of the proposed model's performance with SOTA methods on the CASIA v2 dataset under different protocols. A "-" indicates that no corresponding results were reported. All values are in %.

Protocol	Method	F1-score	Precision	Recall
Protocol-A	BusterNet [163]	67.05	77.38	59.15
	QDL-CMFD [212]	71.11	89.18	66.02
	Proposed	74.97	89.16	69.28
Protocol-B	BusterNet [163]	45.56	55.71	43.83
	QDL-CMFD [212]	53.25	73.35	59.16
	Proposed	66.43	74.32	60.92

For the CASIA v2 dataset, two evaluation protocols are employed for pixel-level assessment, each differing in the way the F1-score is computed. Protocol A calculates precision, recall, and F1-score by aggregating all TP, FP, and FN across the entire dataset. In contrast, Protocol B determines these metrics on a per-image basis and then reports the average scores. Protocol A provides an overall performance evaluation, including non-forged images, whereas Protocol B focuses on a subset of forged images.

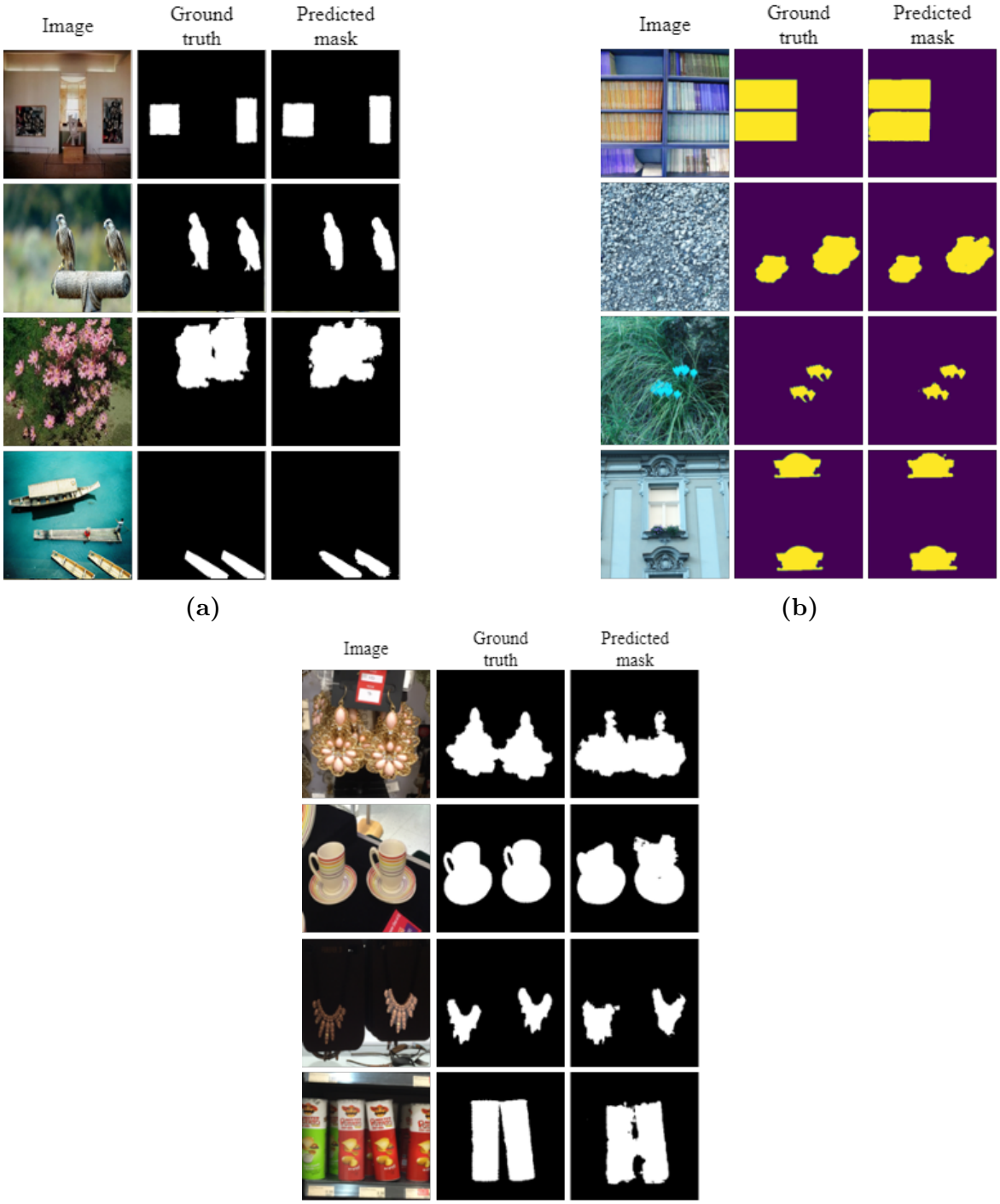


Figure 3.6: Sample results of the proposed model on the (a) CASIA v2 (b) CoMoFoD, and (c) COVERAGE datasets.

Since the F1-score is undefined when TP is zero, Protocol B is more suited for assessing localization performance. Given these distinctions, both protocols are utilized for a comprehensive evaluation.

The results indicate that the proposed model outperforms existing methods under

Table 3.5: Comparison of the proposed model's performance with SOTA methods on the CoMoFoD dataset. A "-" indicates that no corresponding results were reported. All values are expressed as %.

Method	F1-score	Accuracy	Precision	Recall
Liu et al. [92]	77.42	-	78.85	82.02
DOA-GAN [161]	63.05	-	60.38	60.38
UCM-Net [162]	57.00	-	58.00	54.00
Lin et al. [213]	59.40	-	-	-
BusterNet [163]	55.22	-	53.20	57.41
QDL-CMFD [212]	83.14	-	66.46	66.46
Proposed	84.51	99.34	92.30	81.92

Table 3.6: Comparison of the proposed model's performance with SOTA methods on the COVERAGE dataset. A "-" indicates that no corresponding results were reported. All values are in %.

Method	F1-score	Accuracy	Precision	Recall
UCM-Net [162]	73.00	-	69.00	84.00
Lin et al. [213]	35.30	-	-	-
Chen et al. [164]	67.70	-	-	-
Li et al. [214]	72.28	-	80.22	41.76
DS-UNet [215]	67.70	94.80	-	-
PSCC-Net [216]	72.30	-	-	-
Proposed	79.81	98.57	80.60	78.75

both protocols. Its superior performance across multiple datasets, particularly in comparison to other U-Net variants, highlights the significance of the patch-wise attention mechanism and its potential in the field of CMFD.

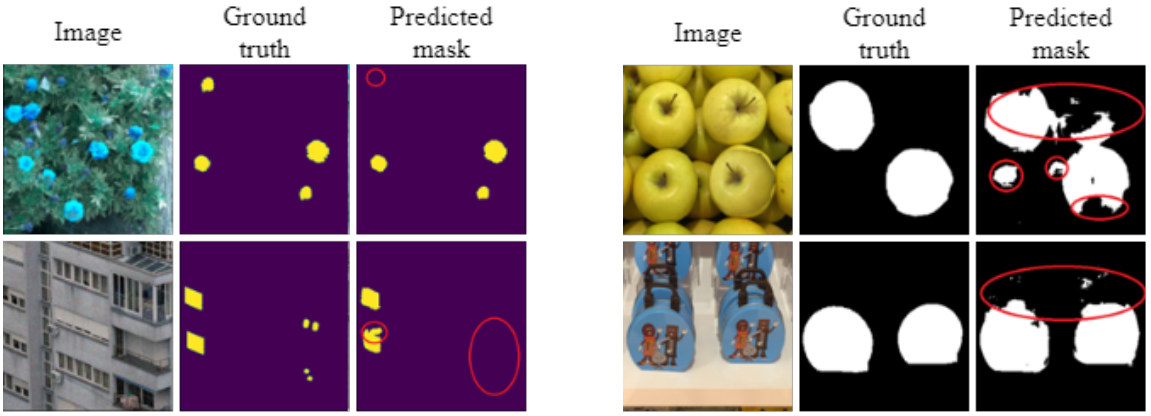
3.1.9 Error Analysis

The model occasionally exhibits reduced performance when regions of interest vary significantly in size (see Figure 3.7). This limitation arises due to the nature of the patches (e.g., 4×4 or 8×8), which may sometimes be smaller than the actual regions of interest. In cases where rows and columns are considered for cosine similarity, variations in scale can make the forged region appear dissimilar to the original, leading

Table 3.7: Performance comparison of the proposed model with SOTA methods on MICC-F600. A "-" indicates that no corresponding results are available. All values are expressed in %.

Method	F1-score	Accuracy	Precision	Recall
Tinnathi et al. [217]	92.75	-	92.45	93.67
Li et al. [214]	91.50	-	97.5	91.80
Proposed	93.96	96.28	93.47	93.95

to model failure. This issue is particularly pronounced when dealing with extremely small regions, as shown in the error cases. Additionally, the bottleneck layer of the model compresses features, potentially resulting in the loss of information related to very small objects. While this impacts scale invariance, addressing the issue by incorporating an attention mechanism in each encoder would introduce significant time complexity and computational overhead.



(a) Missed forged regions.

(b) Segmentation errors in detecting circular forged regions.

Figure 3.7: Some error cases of the proposed model, where the red encircled region highlights the incorrect segmentation produced by the model.

3.2 Discussion

The current chapter addressed the complexities associated with CMFD by proposing an improved and resource-efficient MultiResUNet model. This enhanced model leverages the strengths of multi-resolution analysis and U-Net-based architectures to effectively identify duplicated regions in an image, even under challenging conditions such as homogeneous backgrounds or subtle transformations. To evaluate the effectiveness of the proposed approach, extensive experiments were conducted on four benchmark datasets widely used for CMFD research: CASIA TIDE v2.0, CoMoFoD, COVERAGE and MICC-F600. These datasets present a variety of real-world forgery scenarios, including difficult cases with low contrast between forged and original areas. The proposed method achieved superior accuracy on all datasets, including 89.16% on CASIA v2, 99.34% on CoMoFoD, 98.57% on COVERAGE and 96.28% on MICC-F600, outperforming existing SOTA techniques.

Chapter 4

Deepfake Detection in Images/Videos

The emergence of GANs [7] has marked a turning point in the field of computer vision, enabling remarkable progress in synthetic image and video generation. By training generative and discriminative models in tandem, GANs can create highly realistic data samples that closely resemble those in the original dataset. These capabilities have led to numerous innovative applications, such as cartoon character generation, virtual clothing design, and the synthesis of high-quality visual content.

However, the same technological advancements that make GANs so powerful also introduce significant societal challenges. The ability to generate visually convincing yet entirely artificial media such as fake face generation in images and videos has raised serious security concerns. Synthetic content can be easily used to spread misinformation, falsify evidence, and manipulate public opinion. As a result, detecting manipulated or forged media generated through deep learning techniques has become an urgent and critical area of research.

As GANs architectures continue to evolve, the content they generate becomes increasingly indistinguishable from authentic media, particularly in the context of facial manipulation. The subtle differences between real and fake visuals are often imperceptible to the human eye. This has made automated detection methods essential, many of which rely on identifying minute inconsistencies or artifacts left during the generation process. Previous studies have explored a variety of detection strategies, including the analysis of retina color anomalies [218], irregular blinking patterns [219], inconsistencies across facial regions [220], and other visual artifacts [221, 222]. More recent approaches using CNN-based and transformer-based models have attempted to identify these manipulated regions by learning patterns of tampering. Despite this progress, a major limitation persists, as discussed in Chapter 2. Existing methods often focus on specific facial regions like the eyes, nose, lips, or boundaries, which limits their ability to generalize to diverse manipulations and datasets.

The proliferation of such manipulated media, especially on social platforms, poses a growing threat to society. When the general public consumes falsified content without questioning its authenticity, the consequences can be far-reaching. Although fake media often appears realistic, it typically contains subtle distortions or artifacts that may reveal its synthetic origin.

Objectives of the Chapter

The primary objectives of this chapter is to present effective deepfake image and video detection methods. The specific objectives are:

- Design a deep learning-based model capable of accurately detecting deepfake manipulations in image and video with strong generalization demonstrated through comprehensive evaluation on public datasets.
- Combine local and global facial features to identify inconsistencies across different regions of the face by enhancing the receptive field of the CNN, thereby improving the model's ability to capture subtle manipulation cues.
- Explore the effectiveness of combining deep features with handcrafted features to enhance the detection of deepfake artifacts and improve overall performance.
- Apply hierarchical feature selection on the combined feature set to reduce dimensionality and eliminate redundant features, ensuring a more efficient and discriminative representation for deepfake detection.

To fulfill the stated objectives, this study explores multiple approaches to improve deepfake detection in images and videos by analyzing both global and local facial features. Early experiments [43, 44, 49] evaluate the impact of different color spaces, while later methods incorporate soft attention and hierarchical feature selection to enhance detection. Two effective approaches—**ViXNet** and a **Hierarchical Feature Selection (HFS)**-based method are selected for detailed discussion due to their strong performance and relevance. These methodologies aim to enhance the focus on receptive fields by analyzing both local and global features, while also improving the generalization capability of deepfake detection systems for more accurate identification of forgeries across diverse and unseen datasets.

4.1 Methodology 1: ViXNet - A Deep Learning-based Model for Deepfake Detection

In this study, ViXNet is introduced as the first effective deep learning-based classification approach specifically designed to distinguish real videos from forged ones containing manipulated faces generated using various face-swapping techniques. This approach leverages subtle, nearly imperceptible artifacts that remain in specific facial regions after manipulation. ViXNet detects these inconsistencies by analyzing both

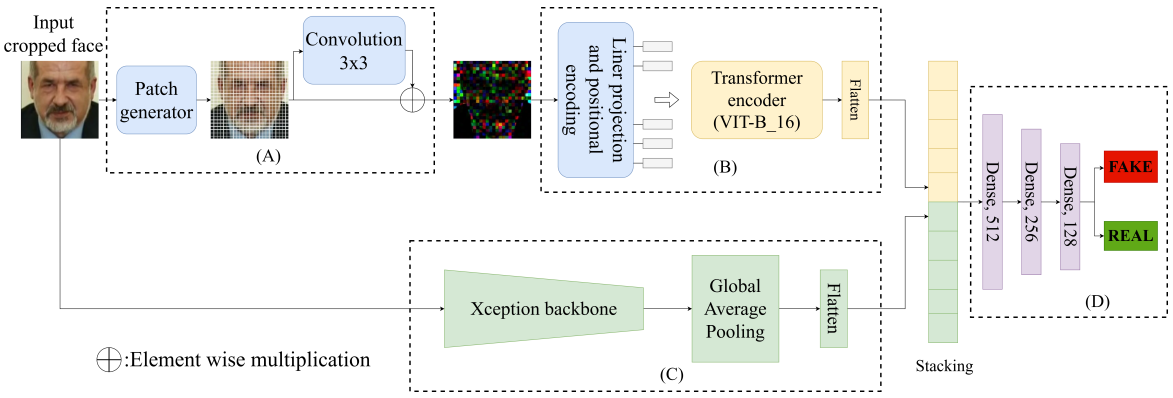


Figure 4.1: The proposed ViXNet model extracts both local and global image features for deepfake video detection. It comprises two primary components: one that first extracts masked local features and then captures global inconsistencies among them, and another that generates global image features. The model consists of the following key modules: (A) a patch-wise self-attention module that generates local features by applying masking to each image patch, (B) a multi-headed self-attention module that detects global inconsistencies among the extracted local features, (C) a CNN backbone responsible for generating global image features, and (D) a feature stacking and classification module that combines intra- and inter-patch features from (B) with spatial global features from (C) to determine whether an image is real or fake.

local and global facial features. It integrates two deep models: one that segments images into patches, applies patch-wise soft attention, and processes them with a Vision Transformer (ViT) [223] to identify inconsistencies, and another Xception network [47] that extracts global spatial features. The overall architecture of ViXNet is illustrated in Figure 4.1, with further details provided in the following sections.

4.1.1 Patch-wise Self-Attention Module

The first module of ViXNet processes an input target image of size $(N \times N)$ by dividing it into M patches of dimension $(P \times P)$, where $\mathbf{M} = \left(\frac{\mathbf{N}}{\mathbf{P}}\right)^2$. Each patch is then passed through a patch-wise self-attention module, which incorporates a masking mechanism. In this process, each patch $\mathbf{I}$ is element-wise multiplied by a weight matrix $\mathbf{W}$ of size $(P \times P)$, producing a masked patch $\mathbf{J}$, as illustrated in Equation 4.1. The weight matrix $\mathbf{W}$ is generated by applying a (3×3) convolution on the patch $\mathbf{I}$, as demonstrated in Block A of Figure 4.1.

$$\mathbf{J}_{x:x+P,y:y+P} = (\mathbf{W}_{i,j} * \mathbf{I}_{i,j}), \quad 0 \leq x, y \leq \mathbf{N} - \mathbf{P}, \quad x \leq i \leq x + \mathbf{P}, \quad y \leq j \leq y + \mathbf{P} \tag{4.1}$$

In this module, each patch is masked using a unique set of trainable weights based

on its position within the image. As a result, the mask is learned differently for various regions, corresponding to different patches. This approach effectively addresses artifacts that appear in specific local areas of a forged face. Once the masked map is generated, it is passed through a transformer to capture global inconsistencies among the masked patches.

4.1.2 Global Self-Attention Module

This module focuses on identifying inconsistencies among the masked patches obtained from the previous stage. Detecting these inconsistencies requires computing the similarity not only within local neighborhoods but also between regions that are spatially distant. To achieve this, a transformer is employed for capturing relationships among inconsistent artifacts localized in the masked patches. The module consists of multiple transformer blocks, each composed of a multi-head self-attention mechanism and a feedforward multilayer perceptron. The multi-head self-attention block generates feature maps that highlight inconsistencies among patches. This is accomplished by converting the masked image patches into linear patch embeddings $X \in \mathbf{R}^{M \times d}$, where **M** represents the number of patches in the image and **d** denotes the embedding size. Positional encoding is then applied in the embedding space to maintain the spatial alignment of patches.

The final embedding matrix is projected using three matrices: $W_Q \in \mathbf{R}^{d \times d_Q}$, $W_K \in \mathbf{R}^{d \times d_K}$, and $W_V \in \mathbf{R}^{d \times d_V}$, which extract feature representations as key (**K**), query (**Q**), and value (**V**), as shown in Equation 4.2, with $d_K = d_Q$ in the case of self-attention.

$$Q = XW_Q, \quad K = XW_K, \quad V = XW_V \tag{4.2}$$

The obtained **K** and **Q** matrices are used to compute the attention score map, which consists of weights applied to the **V** matrix to extract relevant features from the transformed image representation. The attention score map is generated by performing a matrix multiplication of **Q** and the transpose of **K**, followed by a scaling operation with a factor of d_k. A softmax operator is then applied to the resulting score matrix. The final attention output (**A**) is obtained by multiplying the computed score matrix with the **V** matrix, as illustrated in Figure 4.2. Mathematically, this can be expressed as:

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_k}}\right)\mathbf{V} \tag{4.3}$$

This study utilizes the pre-trained transformer model ViT-B_16, which comprises a spatial embedding block followed by position encoding. The architecture includes 12

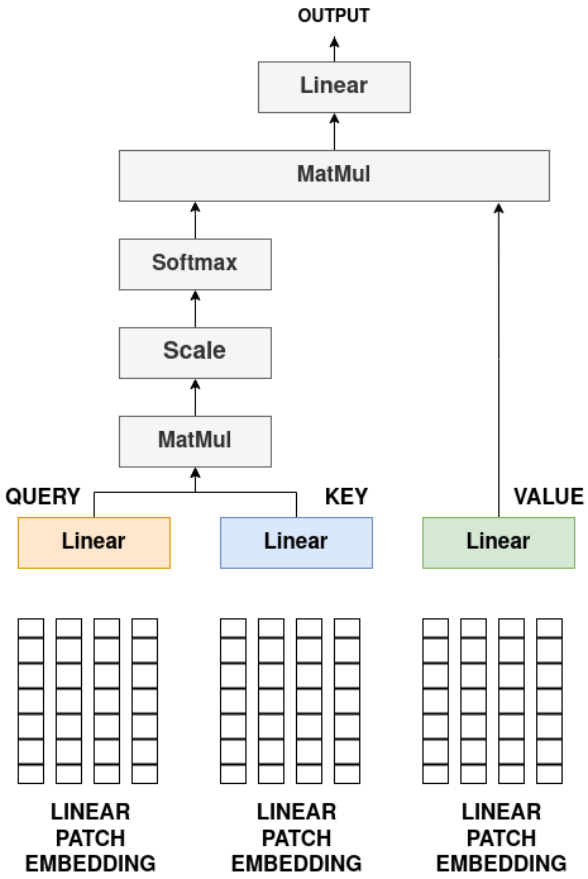


Figure 4.2: A single-headed self-attention unit in transformer blocks helps identify global correlations among patch elements. When multiple such units are stacked in parallel, they form multi-headed self-attention mechanisms.

transformer blocks, making it lighter than other pre-trained models like ViT-L_16 or ViT-L_32. Additionally, the transformer blocks in ViT-B_16 use a patch size of 16×16, which is smaller than the 32×32 patch size in ViT-B_32. This reduction in patch size contributes to a more lightweight architecture overall. The model is trained on the ImageNet dataset, ensuring it learns diverse features compared to models trained on CIFAR-10 or CIFAR-100. This module outputs a feature map, which is subsequently fused with the feature map generated by a backbone CNN, as detailed in the following subsection.

4.1.3 Global Image Feature Extraction

Xception is selected in this study as the global feature extractor due to its proven effectiveness in capturing rich, discriminative features, especially when the training and testing data share similar types of forgery artifacts. Its ability to achieve SOTA prediction accuracy under such conditions makes it highly suitable for deepfake detection tasks where subtle manipulations need to be identified. Architecturally, Xception (Extreme Inception) is a deep CNN that replaces the traditional Inception modules with

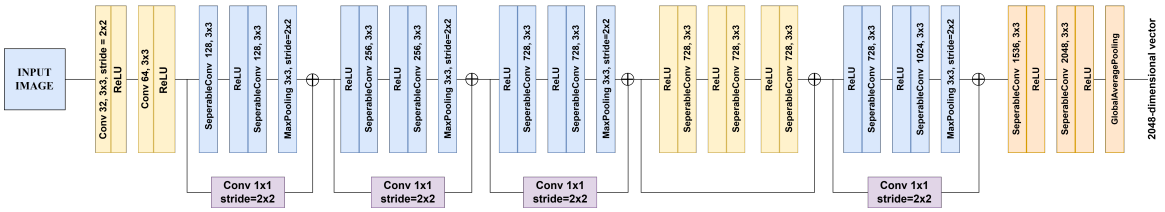


Figure 4.3: The architecture of the Xception model is utilized to extract global feature representations from input cropped face images.

depthwise separable convolutions, enabling efficient modeling of both spatial and cross-channel correlations. These convolutions first operate independently on each channel (depthwise) before combining them (pointwise), allowing the network to disentangle spatial and depth features effectively. Xception includes a 36-layer convolutional base organized into 14 modules, as shown in Figure 4.3. Although it maintains a similar number of parameters as the Inception model, it utilizes them more efficiently, enabling it to extract meaningful patterns across varying spatial and depth scales. This makes it a powerful backbone for computer vision applications and motivates its integration into the proposed framework for robust global feature representation.

4.1.4 Classification Model

The models described in the previous subsections generate two distinct types of features: one capturing global inconsistencies among masked patches to derive intra-inter patch information, and the other representing the spatial global features of the image. In this stage, both feature types are first stacked together, followed by the design of a classifier on top of the stacked features (see Module (D) in Figure 4.1). The classifier consists of a series of fully connected layers with 512, 256, and 128 nodes, respectively, which are used to learn a non-linear function over the stacked features. Rectified Linear Unit (ReLU) activation is applied in these layers to introduce non-linearity. For the final classification layer, a softmax activation function is employed, providing the probability of an image being either fake or real.

4.2 Experimental Results and Discussion

As previously mentioned, this work introduces a deep learning model designed to determine whether a given video contains a manipulated face generated using deepfake techniques. This section begins by detailing the forensic datasets used for experimentation, followed by an explanation of how the train and test datasets are prepared. The training protocols employed are then discussed. To evaluate the model, its performance is assessed using both intra-dataset and inter-dataset experimental setups (details in Subsection 4.2.3) across three publicly available deepfake datasets using the

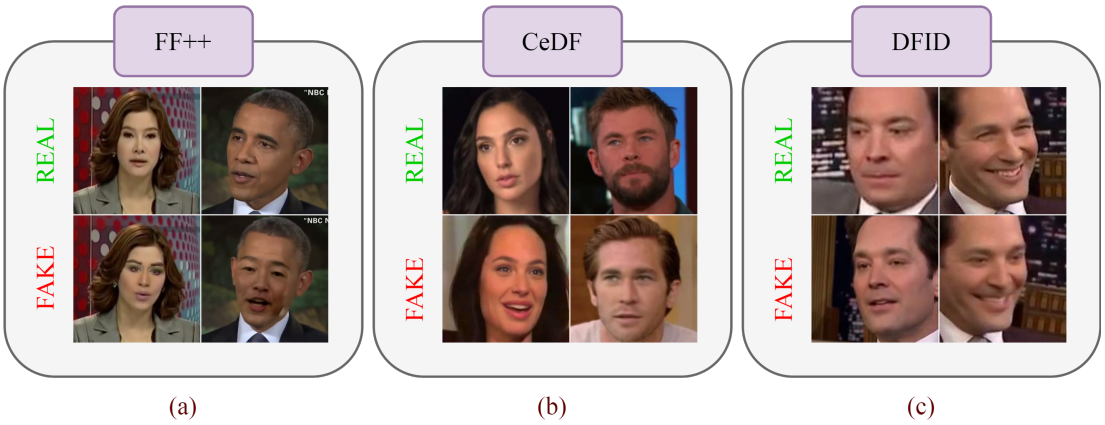


Figure 4.4: Examples of real and forged images (generated using face-swapping techniques) from the following datasets: (a) FF++, (b) CeDF, (c) DFID

metrics discussed in Chapter 3, Section 3.1.3. These setups help validate the generalizability of the proposed ViXNet model. Finally, the model's performance is compared with several SOTA methods. The proposed model is implemented in Python using the TensorFlow framework, and all experiments are conducted on the Google Colab platform, utilizing an Nvidia Tesla K80 GPU with 4992 CUDA cores.

4.2.1 Dataset used for the Proposed Methods

The experiments utilize four widely recognized and publicly available deepfake datasets: FaceForensics++ (FF++) [174], Celeb-DF (V2) (CeDF) [175], Deepfake Image Dataset (DFID) [42] and DeepFake Detection Challenge (DFDC) [224]. Sample images and cropped face images from video frames for each dataset are illustrated in Figure 4.4.

- **FF++** – It is a forensic video dataset containing 1,000 real videos, sourced from 977 YouTube clips, curated by Rössler et al. [174]. Each video features a trackable, predominantly frontal face without occlusions, enabling automated synthetic manipulation to produce realistic forgeries. Four deepfake generation techniques: *Deepfakes, Face2Face, FaceSwap*, and *NeuralTextures* were applied to create 4,000 forged videos, bringing the total dataset size to 5,000 videos (1,000 real and 4,000 fake). Since most synthetic videos on the web are compressed, FF++ provides videos at two H.264 encoding quality levels: C-23 (high quality, HQ) and C-40 (low quality, LQ).

- **CeDF** – It is a large-scale dataset where deepfake videos were created using an enhanced deepfake generation method. It includes 590 real videos and 5,369 manipulated videos, comprising over 2.3 million frames in total. The average duration of each video is approximately 13 seconds, recorded at a standard frame

rate of 30 frames per second. The pristine videos were sourced from publicly available YouTube content featuring 59 celebrities with diverse demographics [175].

- **DFID** – The deepfake image dataset, referred to as DFID, comprises 8,000 forged and 11,809 real face images. These images were extracted from 175 videos sourced from various online platforms, as collected by [42]. All videos were encoded using the H.264 codec at varying compression levels, resulting in forged images that closely resemble authentic ones. The dataset includes only cropped face images, obtained from video frames using the Viola-Jones face detector [225]. To ensure a balanced representation, the number of frames selected for face extraction from each video was made proportional to the variations in camera angles and lighting conditions affecting the target face.

- **DFDC** – First introduced during a Kaggle competition [226] and later released publicly by Facebook AI [224], the DeepFake Detection Challenge (DFDC) dataset is considered one of the most comprehensive third-generation forensics datasets available. The DFDC dataset is pre-partitioned into training, validation, and test sets, facilitating standardized development and evaluation of deepfake detection models. The training set includes 119,154 ten-second video clips featuring 486 subjects, with manipulated videos generated using various models such as DFAE, MM/NN FaceSwap, NTH, FSGAN, and StyleGAN. The validation set comprises 4,000 ten-second clips of 218 subjects. Additionally, the dataset includes a public test set of 5,000 clips and a private test set of 10,000 clips.

4.2.2 Data Preparation

Among the previously mentioned datasets, FF++, CeDF, and DFDC contain face manipulations in video format. In contrast, the DFID dataset consists of individual images, eliminating the need for additional face cropping. Consequently, a preprocessing step is necessary only for the video-based datasets. Initially, the OpenCV Haar Cascade method was used to detect face regions, but it frequently resulted in false positives. To address this, a more robust approach was adopted using the CNN-based Multi-Task Cascaded Convolutional Networks (MTCNN) [227], which offers improved accuracy across diverse video qualities, face alignments, and face sizes. Once the face region is detected, the bounding box is expanded by 10% (or 10 pixels in each direction) to include some background context, and cropped face images are extracted

accordingly. These cropped faces are then resized to (299 × 299) for further use. I-frames are preferred over other types of frames in a test video because they are stored in an uncompressed format and preserve the most color information. The overall pre-processing pipeline is illustrated in Figure 4.5.

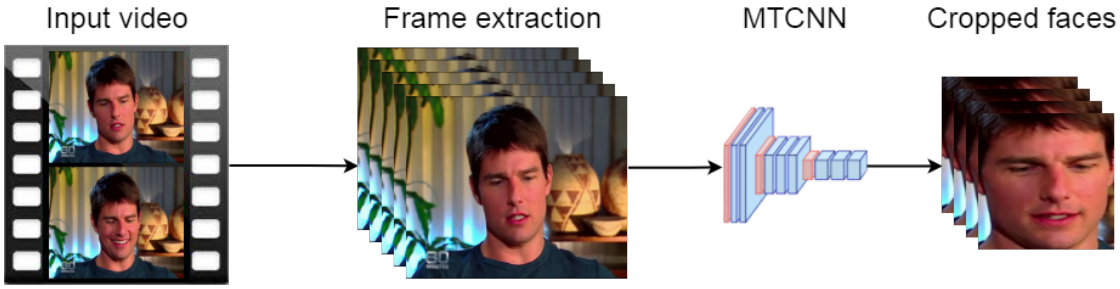


Figure 4.5: Illustration of image data preparation from a video.

For the FF++ dataset, only the Deepfakes subset is used, as the focus is on detecting videos manipulated through face-swapping techniques. The 1000 videos in each class (Deepfakes and Pristine) are split into three subsets: 70% for training, 20% for validation, and 10% for testing. This results in 5876 cropped face images from 1400 training videos, 400 cropped images from 400 validation videos, and 200 cropped images from test videos. Equidistant frames in the temporal space are selected for the training set, ensuring diverse face postures as no two consecutive frames are adjacent. For validation and test sets, only the first frame of each video is considered.

In the CeDF dataset, 5011 videos are used for training, 1000 for validation, and 518 for testing. Notably, the 518 test videos are the ones provided by the dataset authors. Cropped face images for training, validation, and testing are extracted using the same approach as in FF++, resulting in 9152, 1000, and 518 samples, respectively.

For the DFDC dataset, the training set was constructed by randomly selecting one frame from each of 86,166 videos in the training partition, resulting in a total of 86,166 images. Among these, 13,916 images belonged to the real class, while the remaining 72,550 were labeled as fake. To address the class imbalance during training, the real images were over-sampled. For evaluation, one face frame from each video in the public test set comprising 5,000 video clips was used to assess the performance of the model.

4.2.3 Experimental Setups

A robust forensic detection model should be capable of distinguishing forged cases from real ones, regardless of the datasets and forging techniques used. The proposed ViXNet model is designed to achieve this capability. To evaluate its generalizability and robustness, experiments are conducted using two different approaches: the

intra-dataset approach and the inter-dataset approach. In the intra-dataset approach, ViXNet is trained, validated, and tested on a single dataset. Specifically, training is performed on a set denoted as A_{train}, and the best model is selected based on performance on the validation set $A_{validation}$. The selected model is then tested on the test set A_{test}. This approach ensures that the model performs optimally on a given dataset $A_{validation}$. In contrast, the inter-dataset approach involves training ViXNet on a dataset A_{train}, selecting the best model based on a different validation set $B_{validation}$, and then testing it on a separate dataset C_{test}. This procedure evaluates the model on a dataset different from the one used for training, resulting in a hybrid model capable of detecting artifacts across multiple datasets. The results obtained from these experiments are discussed in later sections. Since the forensic datasets—FF++, CeDF, and DFID—are generated using different face-swapping techniques, the performance of ViXNet in the inter-dataset experimental setup demonstrates its effectiveness regardless of the dataset used for training.

4.2.4 Model Parameters

ViXNet is trained as a whole, despite consisting of multiple trainable components. The backbone CNN, Xception, which is responsible for generating global features, is initialized with weights pretrained on the ImageNet dataset. The global self-attention module, ViT-B_16, is also pretrained on ImageNet. Other trainable components, such as the patch-wise self-attention module and the classification module, are initialized with random weights. During training and testing, input images are resized to a fixed dimension of 384×384. The model is trained for 50 epochs with a learning rate of 0.0001. A small learning rate ensures gradual loss reduction, allowing the model to converge to an optimal solution. Since the classification output is binary (real or fake), binary cross-entropy is used as the loss function. The Adam optimizer is employed for model training. The batch size is set to 8, with steps-per-epoch values of 40 for training and 50 for validation.

4.2.5 Results using Intra-dataset Experimental Setup

This section presents the performance analysis of ViXNet and various combinations of its components using an intra-dataset experimental setup. This evaluation provides insights into how the entire model and its individual components perform when test artifacts are known during training. Seven different models (described below), including the complete model and six component-based variations, are considered.

1. Model utilizing only the patch-wise self-attention module, referred to as PWSA. In this model, learned patch-wise masks are multiplied with corresponding

patches to generate local features.

2. Model consisting solely of the pre-trained ViT model, referred to as ViT. Here, the original image patches are fed into the pre-trained ViT model to learn global correlations among the patches.

3. Model employing only the pre-trained Xception model, referred to as Xception. In this case, cropped faces are directly fed into the Xception model to obtain classification results based on global spatial image features.

4. Model combining PWSA and ViT, denoted as PWSA+ViT. This configuration aims to generate features that capture correlations among masked local facial features.

5. Model combining PWSA and Xception, referred to as PWSA+Xception. This setup is designed to generate features based on the spatial structure of locally masked cropped face images.

6. Model integrating ViT and Xception, denoted as ViT+Xception. This configuration seeks to generate features that combine global correlations among various patches of face regions with spatial feature maps obtained from Xception.

7. The complete model, ViXNet.

For clarity, the performance of these seven models is reported in terms of test accuracy and Area Under the receiver-operating characteristic Curve (AUC) score on the DFID dataset (see Table 4.1). The results indicate that the performance of the Xception model is very close to that of the complete model. Additionally, incorporating either ViT or PWSA with the Xception model results in reduced performance. However, when both ViT and PWSA are combined with the Xception model (i.e., ViXNet), performance improves. Overall, ViXNet demonstrates slightly better performance than the Xception model, particularly in scenarios where the training and test sets contain similar artifacts in fake faces. Furthermore, the proposed model exhibits a superior ability to learn imperceptible artifacts compared to the Xception model. ViXNet's performance on each dataset is illustrated in Figure 4.6, while the corresponding confusion matrices are presented in Figure 4.7.

4.2.6 Results using Inter-dataset Experimental Setup

This section presents the performance evaluation of ViXNet and its individual components (as described in subsection 4.2.5) under the inter-dataset experimental setup. Training, validation, and testing are conducted on different datasets to assess the

Table 4.1: Performance of ViXNet and its component-based variations in an intra-dataset experimental setup, where the model is trained, validated, and tested on DFID.

Model	Test Accuracy (%)	AUC Score (%)
PWSA	71.02	71.99
ViT	59.95	67.28
Xception	95.20	98.80
PWSA+ViT	59.95	69.69
ViT+Xception	95.95	98.89
PWSA+Xception	83.47	92.85
ViXNet	95.42	98.93

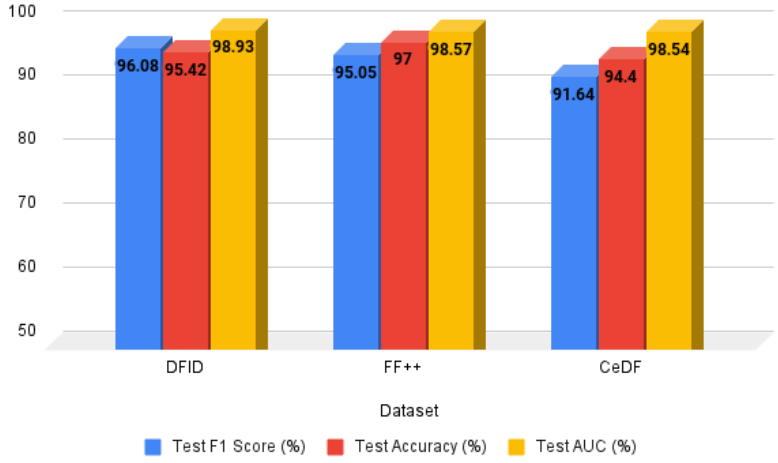


Figure 4.6: Performance of ViXNet across three datasets in the intra-dataset experimental setup.

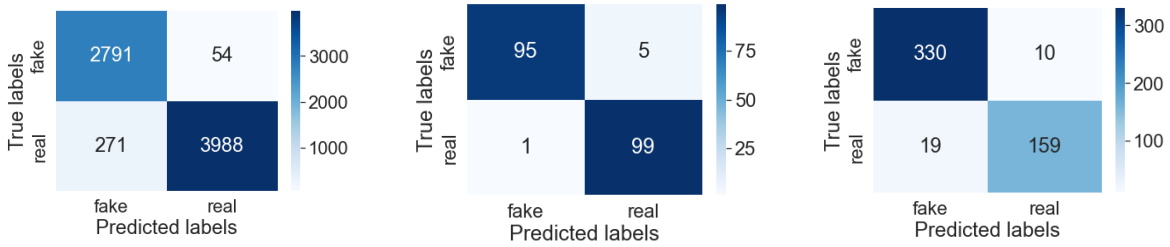


Figure 4.7: Confusion matrices obtained from the intra-dataset experimental setup, where ViXNet is trained, validated, and tested on (a) DFID, (b) FF++, and (c) CeDF datasets.

model's generalizability. The results are summarized in Table 4.3. The experimental protocols followed are detailed in subsection 4.2.3. For simplicity, Table 4.2 reports results for ViXNet and its component-based variations when trained on DFID. The proposed model, ViXNet, consistently outperforms most of its component-based variations in terms of both test accuracy and AUC score. When tested on FF++, ViXNet

demonstrates strong generalization capability, achieving superior classification performance compared to other configurations. However, a relative decline in performance is observed when tested on the CeDF dataset.

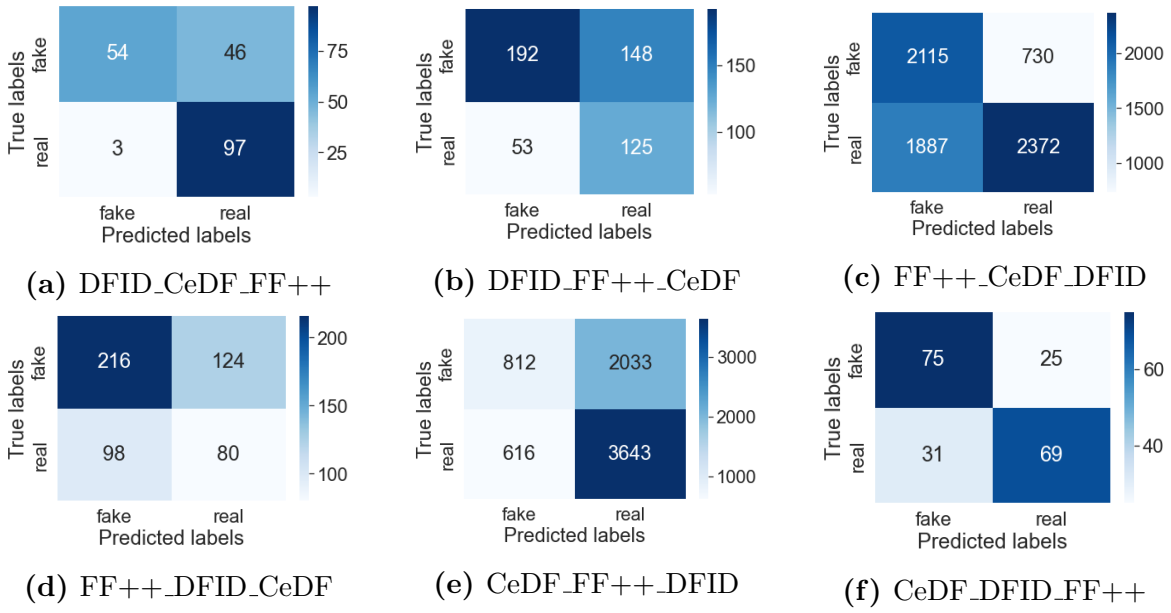
Table 4.2: Performance of ViXNet and its component-based variations under the inter-dataset experimental setup. The results correspond to models trained on DFID.

Validation dataset	Test dataset	Model	Test Accuracy (%)	AUC Score (%)
CeDF	FF++	PWSA	41.70	40.90
		ViT	50.75	53.19
		Xception	66.83	79.18
		PWSA+ViT	50.75	53.18
		ViT+Xception	61.30	69.48
		PWSA+Xception	68.34	75.18
		ViXNet	75.37	83.60
FF++	CeDF	PWSA	50.25	46.82
		ViT	34.36	34.53
		Xception	56.17	58.29
		PWSA+ViT	34.36	34.52
		ViT+Xception	59.84	64.02
		PWSA+Xception	68.53	70.64
		ViXNet	61.19	66.36

Further evaluation is conducted by training on one dataset and testing on a different dataset. The corresponding results are recorded in Table 4.3. These findings highlight the robust performance of the proposed model across various inter-dataset settings. Figure 4.8 presents the confusion matrices obtained during evaluation. A higher occurrence of misclassifications is observed when testing on FF++ (Figure 4.8a and Figure 4.8f), whereas the lowest misclassification rate is noted during testing on the CeDF dataset (Figure 4.8b and Figure 4.8d).

Table 4.3: Performance of ViXNet under the inter-dataset experimental setup across different dataset configurations.

Dataset			Performance (%) on Test Set				
Train	Validation	Test	Accuracy	AUC Score	Recall	Precision	F1-score
DFID	CeDF	FF++	75.49	83.60	97.00	67.83	79.83
DFID	FF++	CeDF	61.19	66.36	70.22	45.78	55.43
FF++	CeDF	DFID	68.90	75.13	55.69	76.46	64.44
FF++	DFID	CeDF	69.30	74.78	45.45	39.21	41.88
CeDF	FF++	DFID	65.08	71.76	85.53	64.13	73.33
CeDF	DFID	FF++	68.00	75.22	69.00	71.13	70.06



(a) DFID_CeDF_FF++ **(b)** DFID_FF++_CeDF **(c)** FF++_CeDF_DFID

(d) FF++_DFID_CeDF **(e)** CeDF_FF++_DFID **(f)** CeDF_DFID_FF++

Figure 4.8: Confusion matrices from different inter-dataset experimental setups. X_Y_Z denotes training on dataset X, validation on dataset Y, and evaluation on dataset Z.

4.2.7 Comparison with Past Methods

This section presents a comparative analysis of the proposed ViXNet model with various SOTA deepfake detection methods, including those introduced by Afchar et al. [42], Chollet et al. [47], Qian et al. [228], Guo et al. [107], Wodajo and Atnafu [35], Mohiuddin et al. [43], Ganguly et al. [44], and Coccomini et al. [229]. To ensure a fair comparison, these methods have been evaluated following the experimental protocols described earlier. The assessment allows for an evaluation of different approaches based on robustness and generalizability. Performance is reported in terms of test accuracy and AUC scores under the specified experimental setup (see subsection 4.2.3) on the FaceForensics image dataset, as detailed in Table 4.4 and Table 4.5. The first column in Table 4.4 and Table 4.5 specifies the experimental protocol used for evaluating the methods. The naming convention follows $Dataset_{train}\_Dataset_{validation}\_Dataset_{test}$. The results indicate that ViXNet achieves superior performance compared to other methods in the intra-dataset experimental setup. For inter-dataset experiments, which assess model generalizability, ViXNet outperforms competing approaches in most cases, achieving SOTA results.

Table 4.4: Test accuracy (in %) of ViXNet compared with various SOTA models under different experimental protocols. The experimental protocols follow the format Train Dataset_Validation Dataset_Test Dataset. M1 - M9 correspond to the methods proposed by Afchar et al. [42], Afchar et al. [42], Chollet et al. [47], Qian et al. [228], Guo et al. [107], Wodajo and Atnafu [35], Mohiuddin et al. [43], Ganguly et al. [44], and Coccomini et al. [229]., respectively.

Experimental Protocol	M1	M2	M3	M4	M5	M6	M7	M8	M9	ViXNet
FF++_FF++_FF++	66.83	64.82	95.48	95.48	77.88	85.93	85.93	84.23	95.98	97
CeDF_CeDF_CeDF	65.64	65.83	89.38	87.06	68.33	65.64	84.56	81.32	90.35	94.40
DFID_DFID_DFID	80.28	78.91	95.21	95.05	84.50	84.71	88.37	82.57	94.76	95.42
FF++_CeDF_DFID	57.28	53.74	71.02	72.71	41.82	62.94	62.53	66.47	63.27	68.90
FF++_DFID_CeDF	51.54	66.41	55.60	63.89	41.31	44.21	63.71	68.04	62.36	69.30
CeDF_FF++_DFID	40.51	45.34	60.36	56.39	57.34	40.05	60.84	67.25	62.80	65.08
CeDF_DFID_FF++	70.35	50.25	64.82	54.27	63.81	50.25	59.80	64.71	68.36	68.00
DFID_CeDF_FF++	63.32	66.33	66.83	75.87	62.81	65.33	65.83	60.24	56.78	75.49
DFID_FF++_CeDF	59.85	64.48	56.17	62.35	55.79	61.78	63.13	59.40	47.30	61.19

Table 4.5: Test AUC scores (in %) of ViXNet compared with SOTA models under various experimental protocols. Experimental protocols follow the format Train Dataset_Validation Dataset_Test Dataset. M1 - M9 correspond to the methods proposed by Afchar et al. [42], Afchar et al. [42], Chollet et al. [47], Qian et al. [228], Guo et al. [107], Wodajo and Atnafu [35], Mohiuddin et al. [43], Ganguly et al. [44], and Coccomini et al. [229]., respectively.

Experimental Protocol	M1	M2	M3	M4	M5	M6	M7	M8	M9	ViXNet
FF++_FF++_FF++	66.81	66.93	98.79	96.52	87.62	96.41	85.92	83.04	99.06	98.57
CeDF_CeDF_CeDF	65.33	64.80	95.59	81.48	78.04	61.07	78.46	80.50	96.63	98.54
DFID_DFID_DFID	70.33	62.53	98.80	95.70	92.11	92.21	89.12	79.51	98.47	98.93
FF++_CeDF_DFID	56.68	50.36	78.17	68.04	40.60	74.24	59.91	63.56	74.71	75.13
FF++_DFID_CeDF	51.33	65.58	58.06	53.89	40.73	50.96	56.43	66.12	63.28	74.78
CeDF_FF++_DFID	45.86	44.37	67.21	62.17	56.80	47.47	52.82	62.36	62.64	71.76
CeDF_DFID_FF++	69.68	51.51	75.19	54.04	71.09	72.37	59.93	63.80	78.84	75.22
DFID_CeDF_FF++	55.60	62.28	79.18	75.96	63.77	73.22	65.70	57.19	82.37	83.60
DFID_FF++_CeDF	54.17	61.65	58.29	59.14	59.17	50.00	54.24	59.01	60.60	66.36

The results in Table 4.5 can be analyzed as follows: (1) ViXNet, utilizing Xception as the global spatial feature extractor, surpasses the performance of the standalone Xception network across all inter-dataset experimental protocols. This improvement highlights the positive impact of the PWSA and Global Self-Attention (GSA) modules in enhancing the base CNN model's generalization capabilities. (2) The superior performance of ViXNet over F3-Net, which primarily focuses on frequency-based analysis for face forgery detection, suggests enhanced robustness to image compression artifacts. (3) ViXNet consistently outperforms the AMTEN network (M5 [107]) in all experimental settings. Unlike AMTEN, which applies global masking, the PWSA module employs localized, region-specific masking, contributing to improved classification accuracy. (4) The Convolutional-ViT model integrates a deep CNN with a vision transformer sequentially, using CNN output as input to the transformer. The superior performance of ViXNet, which decouples CNN and ViT while leveraging independent feature extraction, highlights the advantages of this approach. (5) The method proposed by Coccomini et al. [229] extracts features at multiple scales using different patch sizes, demonstrating effectiveness in deepfake detection. ViXNet's competitive performance further reinforces the importance of multi-scale feature extraction.

4.2.8 Results on the DFDC Dataset

The test set is created by extracting the I-frame from each video in the public test set. The model is trained for 400 epochs using the Adam optimizer with a learning rate of 0.0001. As detailed in Table 4.6, the model achieves an AUC score of 86.32% and an F1-score of 79.06%. Additionally, a comparative evaluation is conducted by replacing the Xception backbone with EfficientNet B0, yielding an AUC score of 86.30% and an F1-score of 78.29%. These results suggest that the proposed approach performs favorably compared to EfficientNet B0, which is used in previous works [229, 230]. Further evaluation is performed using a multi-frame decision approach, where up to 12 random frames are selected from each test video. The final prediction score for a video is computed by averaging the scores of the extracted frames. This approach results in improved performance compared to the single-frame decision method, as shown in Table 4.6. Due to computational constraints, training on the full dataset is not feasible. Only one frame per clip is used, totaling 86,166 frames, whereas [229] uses 220,444 frames. Additionally, training is conducted with 40 batches per epoch, in contrast to the 2,500 batches used by [230]. The performance of various SOTA methods, along with ViXNet, is recorded in Table 4.6, which presents a comparison of AUC and F1-score on the DFDC dataset.

The proposed method outperforms most existing approaches evaluated in this ex-

perimental setup. Notably, Guo et al. [107] utilize a similar approach involving global masking of original face images, but the proposed method demonstrates superior performance, highlighting the effectiveness of a local region-specific masking strategy. Additionally, Ganguly et al. [44] employ an attention-based approach using Xception as the backbone. Since the proposed method also incorporates the Xception backbone, the observed improvements confirm that region-based masking followed by global self-attention outperforms soft attention mechanisms. However, Heo et al. [230] achieve higher AUC and F1-score by leveraging deep CNN feature extraction combined with a vision transformer and distillation techniques. It is also important to note that previous methods utilize every frame from DFDC videos for training, whereas the proposed approach relies on a single frame per video. This significant difference in the number of training samples is reflected in the comparative results on the test dataset.

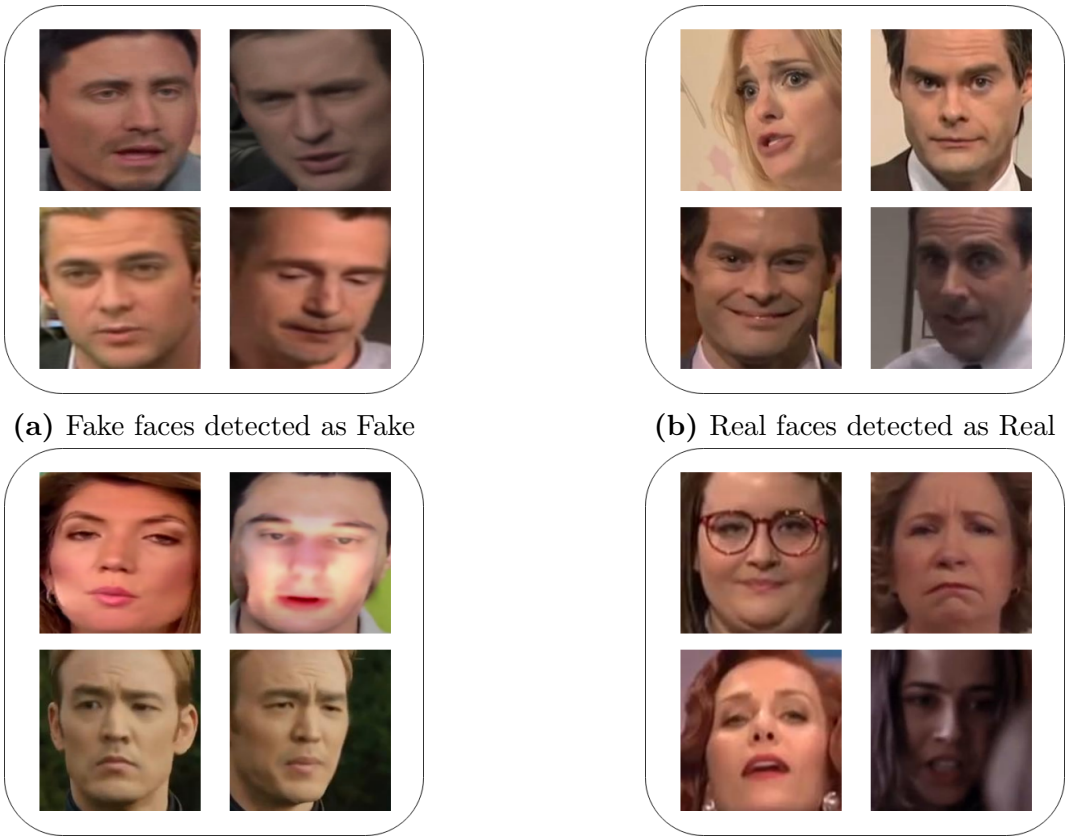
Table 4.6: Comparison of model performance on the DFDC dataset. Methods marked with * were evaluated under the experimental setup described in this work.

Method	#Training Samples	Performance (in %)	
		AUC Score	F1-Score
Heo et al. [230]	$> 1,000,000$	97.80	91.90
Wodajo et al. [35]	162,174	84.30	77.00
Guo et al. [107]*	86,166	74.19	72.83
Coccomini et al. [229]	220,444	92.50	84.50
Mohiuddin et al. [43] *	86,166	71.77	68.00
Ganguly et al. [44]*	86,166	83.59	74.21
ViXNet (Single-frame decision)	86,166	86.32	79.06
ViXNet (Multiple-frame decision, 12 frames)	86,166	90.26	83.18

4.2.9 Error Analysis

This section presents an analysis of ViXNet's performance at the image level. The model is evaluated on the generated test dataset, following an intra-dataset experimental setup (see subsection 4.2.3). The objective is to examine cases where the model correctly classifies images and instances where misclassification occurs. Figure 4.9 illustrates examples of correctly classified images (see Figure 4.9a and Figure 4.9b) as well as misclassified ones (see Figure 4.9c and Figure 4.9d). A common pattern is observed in false positives (see Figure 4.9c), where fake images misclassified as real exhibit lighting variations in the facial region. This suggests that the model, in its current form, does not effectively capture such artifacts. In the case of false negatives (see Figure 4.9d), where real images are misclassified as fake, certain facial expres-

sions or occlusions appear to contribute to the misclassification. This could be due to an insufficient representation of specific facial expressions in the training dataset. Despite these challenges, the model demonstrates the ability to accurately classify forged images with subtle, imperceptible artifacts in the facial region, reaffirming its effectiveness in comparison to existing methods.



(a) Fake faces detected as Fake

(b) Real faces detected as Real

(c) Fake faces detected as Real

(d) Real faces detected as Fake

Figure 4.9: Examples of correctly and incorrectly classified face images by the proposed model. The samples include correctly classified cases—(a) true positives and (b) true negatives—as well as misclassified instances—(c) false positives and (d) false negatives.

4.3 Methodology 2: Hierarchical Feature Selection - based Deepfake Detection using Combined Deep and Handcrafted Features

This study proposes an alternative approach for detecting manipulated face images and videos by integrating handcrafted features with deep learning-based features extracted from cropped face regions, aiming to address the limitations of ViXNet. The deep feature extraction follows the same approach used in ViXNet. Instead of the

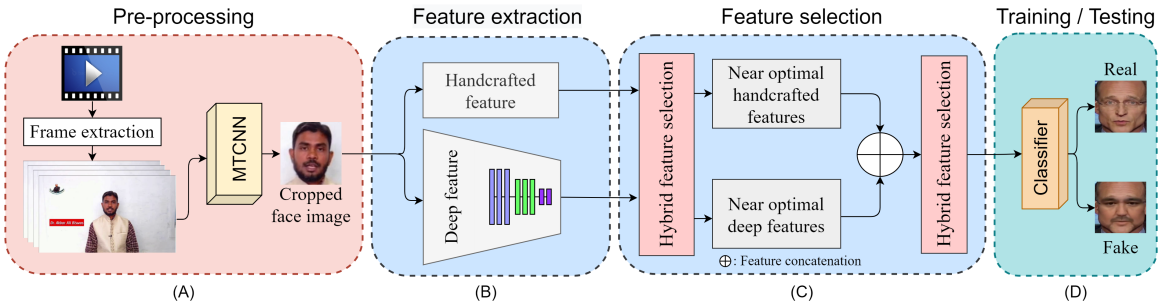


Figure 4.10: The proposed method consists of three stages: (A) Preprocessing, which extracts cropped face images from input videos using MTCNN-based face detection; (B) Feature Generation and Selection, where an HFS technique selects a near-optimal feature set; and (C) Training/Testing, where a classifier determines whether the input video is real or forged.

PWSA path in ViXNet, handcrafted features are employed to capture local inconsistencies, providing a complementary representation alongside the attention-enhanced deep features to improve deepfake detection. Additionally, HFS is performed using a hybrid GWO and VS (GWO-VS) algorithm to eliminate redundant features. The selected near optimal features are then classified using an SVM classifier. The overall workflow of the proposed approach is illustrated in Figure 4.10.

4.3.1 Feature Extraction

This work incorporates both handcrafted and deep learning features as discriminative features, rather than focusing exclusively on one type, such as deep features. The approach is described below.

Handcrafted Feature Extraction: To enable multi-view analysis of the face region as described in [231], each cropped face image is resized from 299 × 299 to 225×225, allowing it to be uniformly divided into 25 patches of size 45×45 pixels each. Daisy features [232] are then extracted from each patch. Daisy, a gradient orientation histogram-based image descriptor, captures both dense and shallow artifacts using a circularly symmetrical kernel and Gaussian weighting. The 'scikit-image' library is used to extract a set of 25 Daisy descriptors, each containing 104 elements per image patch, resulting in a 2600-dimensional feature vector for a single patch.

Feature similarity is estimated using the Euclidean distance (see Equation 4.4) between all extracted patches. In Equation 4.4, E_k represents the Euclidean distance between the i^{th} and j^{th} patches, while $\overrightarrow{P_i}$ and $\overrightarrow{P_j}$ denote the feature vectors for the i^{th} and j^{th} patches, respectively. For each i, the value of j varies from $i+1$ to 25 to compute similarity across all patches. Concatenating all E_k values results in a 300-dimensional feature vector for a single face image. Figure 4.11 illustrates the feature extraction process.

$$E_k = |\overrightarrow{P_i^1} - \overrightarrow{P_j^2}|$$
$$= \sqrt{(P_1^1 - P_1^2)^2 + (P_2^1 - P_2^2)^2 \ldots + (P_n^1 - P_n^2)^2} \tag{4.4}$$

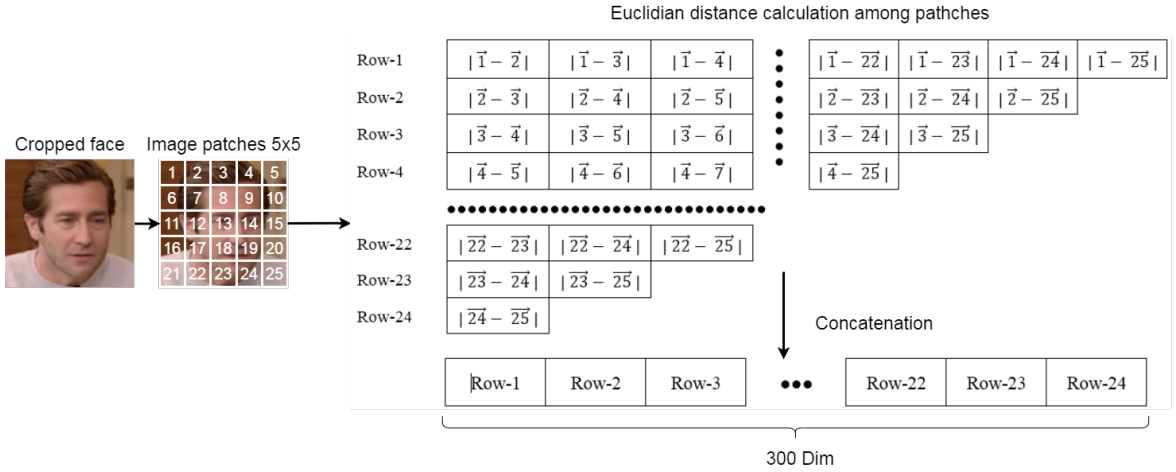


Figure 4.11: Illustration of the handcrafted feature extraction process from a cropped face image. The second image from the left displays the label number $t \in 1, 2, \ldots, n$ assigned to each extracted patch. Daisy features, represented by $\overrightarrow{t}$, are extracted from each patch and used to compute similarity across different patches. Pairwise norm-2 distances (i.e., $|\overrightarrow{i} - \overrightarrow{j}|$, where $i \in 1, 2, \ldots, n-1$ and $i, j \in i+1, 2, \ldots, n$) are calculated and stored in lists named as $Row - i$. The length of each $Row - i$ is $n - i$. These lists are then stacked to form the final feature vector, whose length is given by $(n-1) + (n-2) + \cdots + 1 = \frac{n(n-1)}{2}$. In this case, with $n = 25$, the resulting feature vector has a length of 300.

Deep Feature Extraction: A fine-tuned Xception network [47] is utilized to extract key features from face images, consistent with the approach described in subsection 4.1.3, where the rationale for selecting this architecture is also discussed. This network generates 2048-dimensional deep features from input face images of size 299×299.

4.3.2 Feature Selection

The proposed method introduces an HFS model for feature selection (FS). A hybrid FS approach is designed by leveraging the strengths of GWO [50] and the VS algorithm [51]. The following sections describe the components of the proposed HFS model.

4.3.3 Grey Wolf Optimizer

GWO is inspired by the hunting strategies of grey wolf packs, which exhibit a well-defined hierarchical structure. A typical pack consists of 5-12 wolves, led by an alpha wolf, the primary decision-maker. The alpha's instructions are followed by the rest of

the pack, though decision-making also involves democratic traits, as the alpha considers input from others. Beta wolves, the second in command, relay instructions and provide feedback to the alpha, making them potential future leaders. Delta wolves rank below betas but hold authority over omega wolves, who occupy the lowest position in the hierarchy. Another key aspect of grey wolf behavior is their hunting strategy, which consists of the following phases:

- Tracking, chasing, and approaching the prey.
- Pursuing, encircling, and harassing the prey until it stops moving.
- Attacking the prey.

For feature selection, a solution can be represented as follows: $\overrightarrow{F} = [f_0, f_1, \ldots, f_N]$, where N denotes the total number of features, and f_i (where, $i = 1, 2, \ldots, N$) represents the i^{th} feature in the feature set. Each f_i can take a value of either 0 or 1, indicating exclusion or inclusion of the i^{th} feature in the final feature subset. At any iteration, a candidate solution representing the position of a wolf in the search space is denoted by $\overrightarrow{W}$. The positions of the α, β, and δ wolves are represented as $\overrightarrow{W_\alpha}$, $\overrightarrow{W_\beta}$, and $\overrightarrow{W_\delta}$, respectively. The position update mechanism for wolves in the pack follows the mathematical model given in Equation 4.5 and Equation 4.6:

$$\overrightarrow{D_\alpha} = \overrightarrow{k1} \cdot \overrightarrow{W_\alpha} - \overrightarrow{W}, \quad \overrightarrow{D_\beta} = \overrightarrow{k2} \cdot \overrightarrow{W_\beta} - \overrightarrow{W}, \quad \overrightarrow{D_\delta} = \overrightarrow{k3} \cdot \overrightarrow{W_\delta} - \overrightarrow{W} \tag{4.5}$$

$$\overrightarrow{W_1} = \overrightarrow{W_\alpha} - \overrightarrow{A_1} \cdot \overrightarrow{D_\alpha}, \quad \overrightarrow{W_2} = \overrightarrow{W_\beta} - \overrightarrow{A_2} \cdot \overrightarrow{D_\beta}, \quad \overrightarrow{W_3} = \overrightarrow{W_\delta} - \overrightarrow{A_3} \cdot \overrightarrow{D_\delta} \tag{4.6}$$

The parameters A_i and k_i ($i \in \{1, 2, 3\}$) in Equation 4.5 and Equation 4.6 are defined by Equation 4.7.

$$A_i = 2 \cdot \overrightarrow{a} \cdot \overrightarrow{r_i^1} - \overrightarrow{a}, \quad k_i = 2 \cdot \overrightarrow{r_i^2} \tag{4.7}$$

In Equation 4.7, $\overrightarrow{r_i^1}$ and $\overrightarrow{r_i^2}$ are two random vectors, while the components of $\overrightarrow{a}$ are linearly decreased over the iterations from 2 to 0. The general form of $\overrightarrow{X} \cdot \overrightarrow{Y}$ is given in Equation 4.8, where X and Y represent two agents.

$$\overrightarrow{X} \cdot \overrightarrow{Y} = [X_0 \cdot Y_0, X_1 \cdot Y_1, \ldots, X_N \cdot Y_N] \tag{4.8}$$

Suppose $\overrightarrow{W}$ is represented as $[w^1, w^2, \ldots, w^{N-1}]$. The possible position of the wolf in the next iteration is determined using Equation 4.9.

$$\overrightarrow{W}(t+1) = Func\left(\frac{\overrightarrow{W_1} + \overrightarrow{W_2} + \overrightarrow{W_3}}{3}\right) \tag{4.9}$$

In Equation 4.9, $Func()$ represents a transfer function that maps a continuous value to a discrete binary value. The fitness values of the newly generated and previous positions of a wolf (i.e., candidate solutions) are then evaluated. If the fitness value of the new position is superior to the previous one, the position of the wolf is updated.

4.3.4 Vortex Search Algorithm

The VS algorithm, introduced by Doğan and Ölmez [51], is a single solution-based meta-heuristic optimization algorithm. Suppose $S(\overrightarrow{V})$ represents the set of candidate solutions generated from the current solution $\overrightarrow{V}$. In the replacement phase, a suitable solution (denoted as $\overrightarrow{V'} \in S(\overrightarrow{V})$) is selected from the generated candidate solutions, potentially replacing the current solution. This process continues until the termination condition is satisfied. The phases of the VS algorithm are as follows:

1. Generating the initial solution
2. Generating the set of candidate solutions
3. Ensuring all candidate solutions lie within the search space
4. Replacing the current solution with the best candidate solution if it outperforms the existing one
5. Decreasing the radius of the search space

An initial solution (denoted as $\overrightarrow{V}$) is determined using Equation 4.10 and Equation 4.11.

$$\overrightarrow{V} = [V_1, V_2, \ldots, V_N] \tag{4.10}$$

$$V_i = \frac{uLim_i + lLim_i}{2} \tag{4.11}$$

In Equation 4.11, $uLim_i$ and $lLim_i$ denote the maximum and minimum possible values in the i^{th} dimension, respectively. Also, $i \in \{1, 2, \ldots, N\}$, where N represents the dimension of a single solution, corresponding to the feature set's dimension. Next, a set of neighboring solutions (denoted as $S(\overrightarrow{V})$) is generated randomly using the Gaussian distribution. The multivariate Gaussian distribution is formulated in Equation 4.12.

$$p(\overrightarrow{V}|\overrightarrow{x},\Sigma) = \frac{1}{\sqrt{(2\pi)^N|\Sigma|}} \exp\left\{-\frac{1}{2}(\overrightarrow{x}-\overrightarrow{V})^T\Sigma^{-1}(\overrightarrow{x}-\overrightarrow{V})\right\} \tag{4.12}$$

Here, $\overrightarrow{x}$ represents a randomly generated N-dimensional vector. For a spherical distribution, Σ is defined using Equation 4.13.

$$\Sigma = \sigma^2 \cdot [I]_{N\times N} \tag{4.13}$$

In Equation 4.13, $[I]_{N\times N}$ is an $N\times N$ identity matrix, and σ^2 represents the variance of the distribution. The initial standard deviation is computed using Equation 4.14.

$$\sigma_0 = \frac{\max(uLim) - \min(lLim)}{2} \tag{4.14}$$

The parameter σ_0 serves as the distribution radius. The initial phases of this algorithm require weak locality for enhanced exploration ability; therefore, the maximum possible distribution radius is used. Before replacing the center of the distribution (i.e., the current solution), all candidate solutions lying outside the search space must be adjusted using Equation 4.15.

$$v_i' = \begin{cases} v_i', & \text{if } lLim_i \le v_i' \le uLim_i \\ \text{rand}(uLim_i - lLim_i) + lLim_i, & \text{otherwise} \end{cases} \tag{4.15}$$

In Equation 4.15, v_i' represents a randomly generated candidate solution. In the next step, if the best candidate solution outperforms the current solution based on the fitness score, the existing solution is updated to the best candidate solution. Subsequently, the algorithm gradually decreases the search space radius by reducing the standard deviation, as described by Equation 4.16 and Equation 4.17.

$$\sigma_t = \sigma_0 \cdot gammaincinv(x, \alpha_t) \tag{4.16}$$

$$\alpha_t = \alpha_0 - \frac{iter}{TotalIter} \tag{4.17}$$

In Equation 4.16, $gammaincinv()$ denotes the incomplete inverse gamma function [233].

4.3.5 Proposed Hybrid FS Method

Inspired by the work [234], a single-solution-based meta-heuristic, called the VS algorithm, is utilized to address the premature convergence issue of GWO. The VS algorithm assists in finding improved positions for the wolves at each iteration. The

entire process is described through Equation 4.18, Equation 4.19, and Equation 4.20. Equation 4.18 defines N candidate solutions, while Equation 4.19 describes the selection of the best candidate solution.

$$Set(\overrightarrow{w_i}) = \{\overrightarrow{S_1}, \overrightarrow{S_2}, ...\overrightarrow{S_k}\}, \quad k = 1, 2, \ldots, N \tag{4.18}$$

$$\overrightarrow{W_i}(t+1) = \text{Best}(Set(\overrightarrow{W_i})) \tag{4.19}$$

$$p(\overrightarrow{w}|\overrightarrow{x}, \Sigma) = \frac{1}{\sqrt{(2\pi)^N|\Sigma|}} \exp\left\{-\frac{1}{2}(\overrightarrow{x} - \overrightarrow{w})^T\Sigma^{-1}(\overrightarrow{x} - \overrightarrow{w})\right\} \tag{4.20}$$

Equation 4.20 represents the multivariate Gaussian distribution, where $\overrightarrow{w}$ is the current solution, $\overrightarrow{x}$ is a random N-dimensional vector, and Σ is the covariance matrix. The value of Σ can be computed using Equation 4.13 and Equation 4.14. Over multiple iterations, the search space is progressively reduced using Equation 4.16 and Equation 4.17. The flowchart illustrating the GWO-VS algorithm is presented in Figure 4.12. The time complexity of the GWO-VS algorithm is approximately $\mathcal{O}(P_G \cdot P_V \cdot iter_{max} \cdot j_{max})$, where P_G, P_V, $iter_{max}$, and j_{max} denote the population size of GWO, the population size of VS, the maximum number of iterations for GWO, and the maximum number of iterations for VS, respectively. A detailed complexity analysis and the pseudocode of the GWO-VS algorithm are presented in the Appendix.

4.3.6 Fitness Function

A fitness function is employed to evaluate the performance of a candidate solution. In this approach, the fitness function is designed to balance the trade-off between the number of selected features (denoted as $N_{selected}$) and classification accuracy (denoted as ACC), as defined in Equation 4.21. Here, ACC is measured on a validation dataset using an MLP classifier.

$$fitness(\overrightarrow{w}) = wt * ACC + \frac{N_{selected}}{N} * (1 - wt) \tag{4.21}$$

In Equation 4.21, $wt \in [0, 1]$ (in this case, $wt = 0.90$), and $1 - wt$ represent the weightages assigned to recognition accuracy and the number of selected features for computing the fitness score of a candidate solution. The symbol '*' represents the arithmetic multiplication operator.

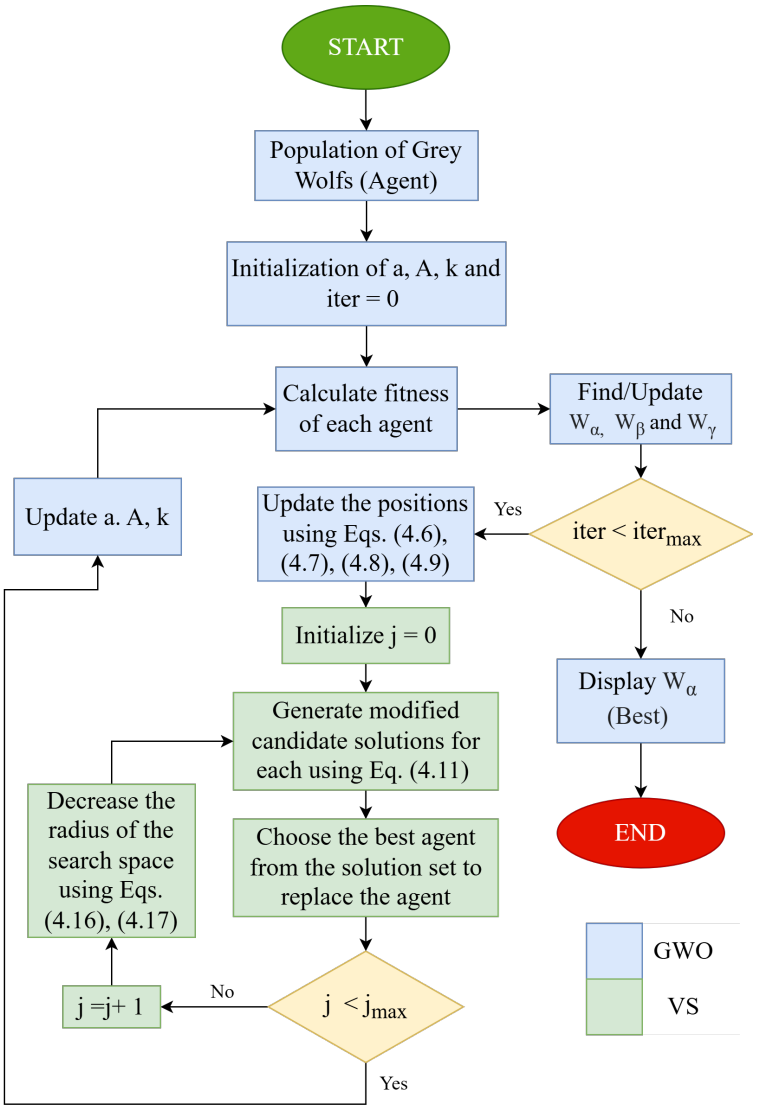


Figure 4.12: Flowchart of the proposed hybrid feature selection technique.

4.3.7 Transfer Function

Since feature selection (FS) is a binary optimization problem, the solution vector must contain only 0s and 1s, where 0 and 1 indicate the exclusion and inclusion of a corresponding feature in the selected feature set, respectively. However, due to the mathematical operations involved in GWO, the values cannot be inherently binary. To ensure that the output remains within the required range, a transfer function is applied to each agent. Here, an S-shaped sigmoid transfer function is used, formulated as follows:

$$T(x) = \frac{1}{1 + e^{-x}} \tag{4.22}$$

$$X^d(t) = \begin{cases} 1, & \text{if } rnd < T(X^d(t)) \\ 0, & \text{if } rnd \geq T(X^d(t)) \end{cases} \tag{4.23}$$

In Equation 4.23, $rnd \in (0, 1)$ is a randomly generated number.

4.3.8 Hierarchical Feature Selection

In the HFS model, multiple types of features are combined to construct a near-optimal feature subset for the classification task. The process of combining and selecting features is carried out in a hierarchical manner. Initially, the proposed hybrid FS technique is applied separately to the extracted handcrafted features (Feature_H in Figure 4.13) and deep learning features (Feature_D in Figure 4.13). This step aims to eliminate redundant features from each type of feature, resulting in the near-optimal feature sets Feature_1 and Feature_2, derived from Feature_H and Feature_D, respectively, as illustrated in Figure 4.13. Next, the obtained near-optimal feature subsets are concatenated, leading to the formation of a new feature subset (Feature_3 in Figure 4.13). This combined feature set (Feature_3) may still contain some irrelevant features [235]. Therefore, the proposed hybrid FS is applied once again to the combined feature set, yielding the final feature subset (Feature_4 in Figure 4.13), which is then used for classification. Figure 4.13 also presents the commonly used FS strategy, where feature concatenation is performed first, followed by the application of the FS method to the combined feature set (Feature_5 in Figure 4.13), producing the near-optimal feature subset (Feature_6 in Figure 4.13).

4.4 Results and Discussion

The proposed model is evaluated on three widely used video datasets: CeDF, FF++ and the DFDC dataset, as described earlier in subsection 4.2.1 of this chapter.

4.4.1 Experimental Setup

Daisy features are extracted using the scikit-image library with the following parameters: $radius = 13$, $rings = 2$, $histograms = 6$, and $orientations = 8$. For fine-tuning the Xception model, the settings used are similar to those of ViXNet model (see subsection 4.2.4). For implementing the GWO-VS model, the parameter values are selected as suggested in [50] and [51] for the GWO and VS algorithms, respectively. For other meta-heuristic algorithms used for comparison, the parameter values are taken from [236].

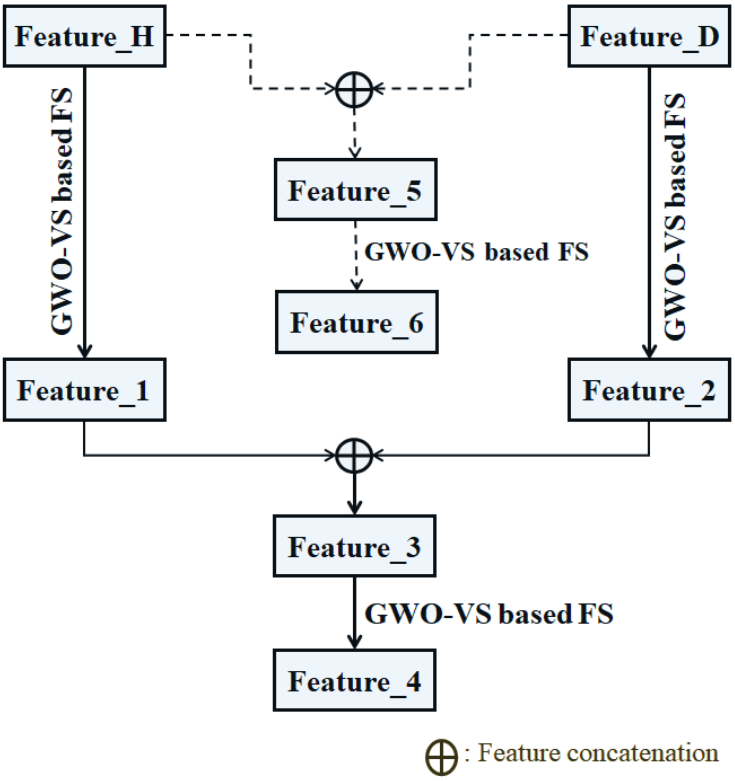


Figure 4.13: The steps involved in generating feature vectors in the proposed HFS model are represented using solid-lined arrows, whereas the steps corresponding to the non-HFS stages are depicted with dashed-lined arrows. Feature_4 and Feature_6 represent the final near-optimal feature subsets obtained through the HFS and non-HFS approaches, respectively.

4.4.2 Experimental Results

This section describes the performance of the proposed HFS-based deepfake detection method. The performance evaluation of the HFS method considers three parameters: AUC, test accuracy, and the number of selected features. The proposed HFS model is executed 10 times on both datasets to analyze the effect of randomness in meta-heuristic results. The results are presented in Figure 4.14. From the results, it is evident that in most cases, test accuracy of 98.00% and 97.30% is achieved on the FF++ and CeDF datasets, respectively. The detailed experimental outcomes are provided in Table 4.7 and Table 4.8 for the CeDF and FF++ datasets, respectively. These results contribute to understanding the performance of the entire model as well as its individual components (see Figure 4.13). As observed from Table 4.7, the test accuracy and AUC score obtained on CeDF using Feature_4 (derived through the HFS method) are 97.30% and 99.35%, respectively, which represent the best performance among all feature sets. Similarly, the HFS model achieves the highest test accuracy and AUC score on the FF++ dataset (see Table 4.8), with values of 98.00% and 99.16%,

respectively. By closely analyzing the results in Table 4.7 and Table 4.8, the following conclusions can be drawn:

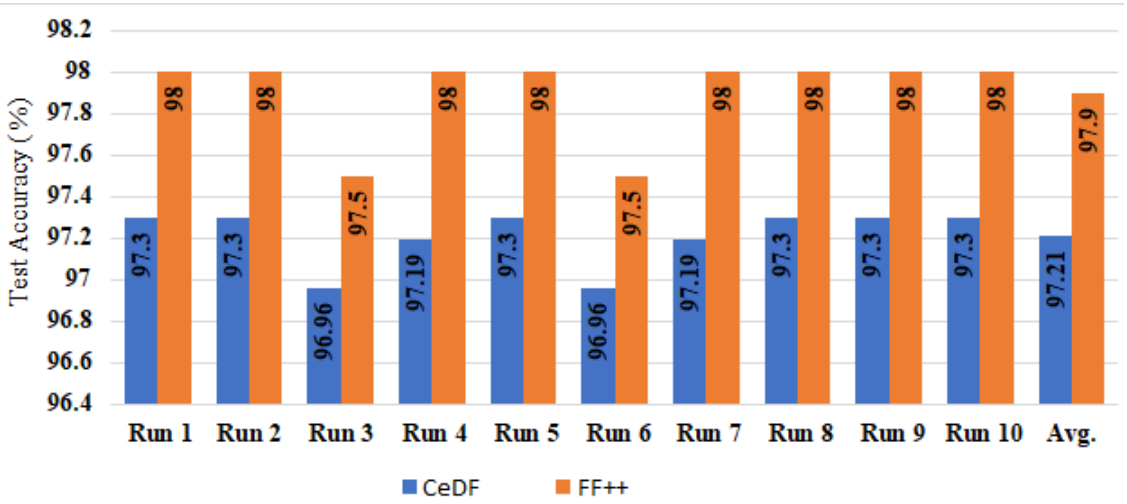


Figure 4.14: Test accuracies of the HFS-based deepfake detection method across ten independent executions.

- When comparing the performance of Feature_D and Feature_H, Feature_D demonstrates superiority. However, concatenating these two features (i.e., Feature_5 in Figure 4.13) results in improved performance. These findings indicate that incorporating handcrafted features representing multi-view analysis alongside deep learning features is beneficial.

- In nearly all cases, the application of the proposed hybrid FS (i.e., GWO-VS) leads to improved performance and reduced feature dimensionality, demonstrating the effectiveness of the hybrid FS technique.

- Most existing methods, regardless of their application, generate the final feature vector by concatenating multiple features (i.e., Feature_5 in Figure 4.13) before classification. On both datasets, utilizing HFS for feature selection-aided classification proves to be more advantageous than this conventional approach. The proposed HFS not only reduces the feature dimension by 88.50% and 87.35% for CeDF and FF++ datasets, respectively, but also enhances performance (CeDF: test accuracy improvement = 0.58%, AUC score improvement = 0.59%; FF++: test accuracy improvement = 1.00%, AUC score improvement = 0.55%).

Table 4.7: Performance of the HFS model and its components on the CeDF dataset (see Figure 4.13). The results under Feature_4 and Feature_6 correspond to HFS-based and non-HFS-based feature selection, respectively. Max, Min, Avg, and SD denote maximum, minimum, average, and standard deviation, respectively.

Features	**Performance Metrics**											
	Test Accuracy (in %)				**AUC Score (in %)**				**# Selected Features**			
	Max	**Min**	**Avg.**	**SD**	**Max**	**Min**	**Avg.**	**SD**	**Max**	**Min**	**Avg.**	**SD**
Feature_H	65.64	64.76	65.16	0.299	73.49	73.07	73.32	0.157	300	300	300	0.00
Feature_D	95.37	94.68	95.01	0.249	98.88	97.92	98.51	0.394	2048	2048	2048	0.00
Feature_1	65.83	65.02	65.56	0.253	71.60	70.95	71.29	0.252	91	76	81.67	4.99
Feature_2	95.14	94.78	95.05	0.129	99.02	98.27	98.80	0.271	722	703	712	6.40
Feature_3	96.91	96.33	96.65	0.211	98.14	97.60	97.96	0.180	813	780	795.5	11.43
Feature_4	97.30	96.83	97.19	0.181	99.35	98.26	98.92	0.405	270	240	257	10.55
Feature_5	96.72	95.75	96.38	0.343	98.76	98.00	98.38	0.310	2348	2348	2348	0.00
Feature_6	96.91	95.95	96.45	0.374	98.14	98.10	98.21	0.136	830	790	812.33	14.53

Table 4.8: Performance of the HFS model and its components on the FF++ dataset (see Figure 4.13). The results under Feature_4 and Feature_6 correspond to HFS-based and non-HFS-based feature selection, respectively. Max, Min, Avg, and SD denote maximum, minimum, average, and standard deviation, respectively.

Features	**Performance Metrics**											
	Test Accuracy (in %)				**AUC Score (in %)**				**# Selected Features**			
	Max	**Min**	**Avg.**	**SD**	**Max**	**Min**	**Avg.**	**SD**	**Max**	**Min**	**Avg.**	**SD**
Feature_H	63.5	62.5	62.90	0.374	70.22	69.84	70.04	0.148	300	300	300	0.00
Feature_D	96.00	95.50	95.70	0.245	98.34	97.8	98.13	0.211	2048	2048	2048	0.00
Feature_1	71.00	70.00	70.55	0.350	76.22	75.56	75.96	0.264	97	90	95	2.89
Feature_2	96.50	95.50	96.10	0.374	97.83	96.78	97.48	0.361	659	640	643.33	11.47
Feature_3	97.50	96.50	97.25	0.335	98.68	98.10	98.46	0.181	756	720	737.17	14.63
Feature_4	98.00	97.50	97.70	0.245	99.16	98.33	98.89	0.273	247	219	234	10.52
Feature_5	97.00	96.00	96.55	0.350	98.61	97.75	98.24	0.340	2348	2348	2348	0.00
Feature_6	97.00	96.50	96.80	0.245	98.68	98.20	98.44	0.162	714	690	703.83	9.39

- In general, methods employing feature selection for classification have followed a non-HFS approach, generating Feature_6 as shown in Figure 4.13. On both datasets, HFS-based feature selection proves to be more beneficial than the non-HFS FS approach. The proposed HFS selects 3.07 and 2.71 times fewer features than Feature_6 for CeDF and FF++ datasets, respectively, while also enhancing performance (CeDF: test accuracy improvement = 0.39%, AUC score improvement = 0.36%; FF++: test accuracy improvement = 1.00%, AUC score improvement = 0.48%) over the single-stage FS approach.

- A comparison of the performances obtained using Feature_3 and Feature_4 (see Figure 4.13) reveals minimal performance improvements across both datasets. However, a significant reduction in feature dimensionality is observed. The dimension of Feature_4 is 3.01 and 3.06 times smaller compared to Feature_3 for CeDF and FF++ datasets, respectively. These results suggest that applying the proposed hybrid FS on Feature_3 is also beneficial in terms of the final feature set's dimensionality and the model's training cost (see Figure 4.15). Figure 4.15 further illustrates that the model with reduced features is trained nearly 2.5 times faster.

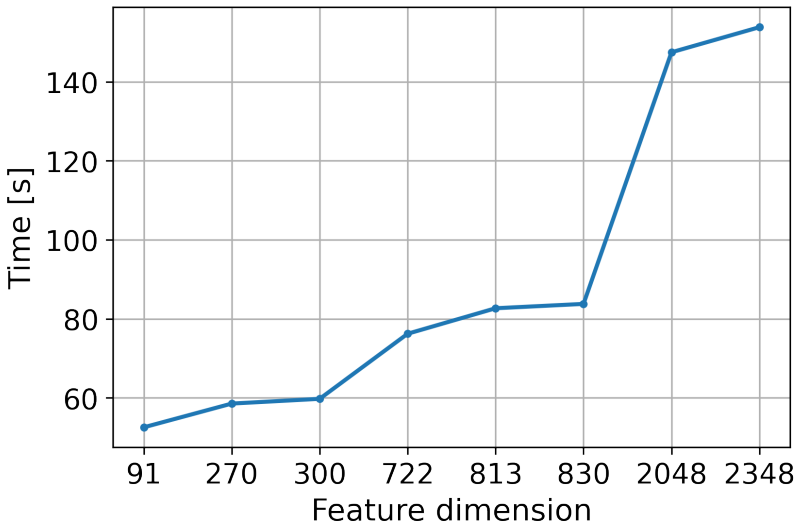


Figure 4.15: Training time as a function of the number of features for the CeDF dataset.

4.4.3 Inter-Dataset Generalization Capability of Selected Features

In this section, another set of experiments is conducted to estimate the generalization capability of the proposed HFS model. For these experiments, the HFS technique is

first applied to the first dataset to obtain the indices of the selected features. Using these selected feature indices, near-optimal features are then extracted from all samples of the second dataset for classification. For example, the selected features obtained by applying HFS to the CeDF dataset (see Table 4.7) are extracted from the FF++ dataset using the indices of selected features, and classification is performed using only those features. The results are presented in Table 4.9. A similar approach is followed to record classification performances on the CeDF dataset, as shown in Table 4.9. These results indicate that the features selected using the proposed HFS technique are more effective compared to the original features, at least for the datasets used in this study.

Table 4.9: Performance evaluation when HFS is applied to one dataset to obtain the near-optimal feature subset and the indices of selected features, which are then used to extract only the selected features from another dataset for classification. Case 1 represents the scenario where HFS is applied to FF++ to obtain feature indices, and classification is performed using the selected features extracted from CeDF. Conversely, Case 2 represents the scenario where HFS is applied to CeDF, and classification is conducted using the selected features extracted from FF++.

Parameter	Feature_D	Feature_H	Feature_5	Feature_6	Feature_4
Case 1					
Test Accuracy (in %)	95.37	65.64	96.72	96.72	97.30
AUC Score (in %)	98.88	73.49	99.35	98.99	99.64
Number of Features	2048	300	2348	804	297
Case 2					
Test Accuracy (in %)	96.00	63.50	97.00	96.50	97.50
AUC Score (in %)	98.34	70.22	97.83	98.68	99.22
Number of Features	2048	300	2348	830	270

4.4.4 Inter-Dataset Generalization Capability of the Proposed HFS Method

A set of experiments is conducted to evaluate the generalization capability of the proposed HFS method. For this purpose, experiments are performed in a inter-dataset setup, following the approach used in [43,44,175]. The results, along with comparisons to SOTA methods, are presented in Table 4.10. In Table 4.10, CeDF_FF++ refers to the experiment where the proposed HFS-based deepfake detection model is trained on CeDF and tested on the FF++ dataset, while FF++_CeDF represents the reverse scenario. Notably, the performances of SOTA methods are evaluated within this experimental setup. The results indicate that the proposed method performs effectively in a inter-dataset setting, demonstrating superior generalization compared to existing SOTA approaches.

Table 4.10: Performance of the proposed model on a inter-dataset experimental setup.

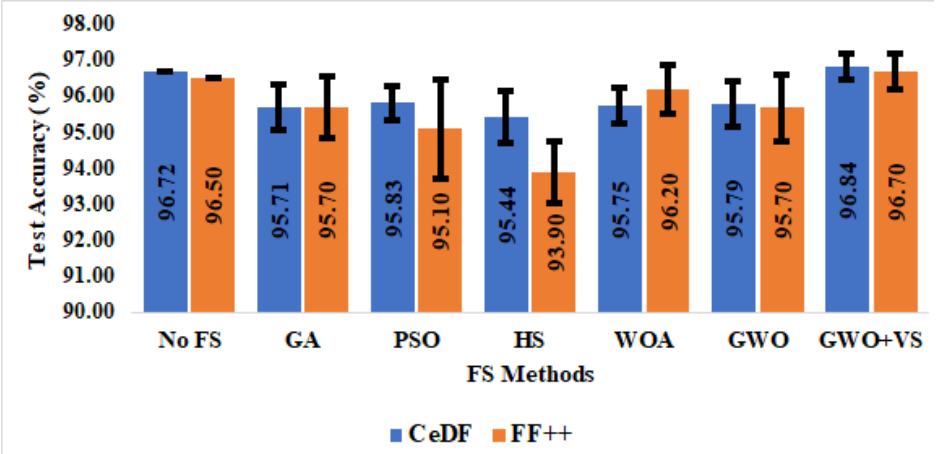
Exp.	Methods	Test Accuracy (in %)	AUC Score (in %)
CeDF_FF++	Li et al. [175]	64.50	75.19
	Afchar et al. [42]	50.50	51.51
	Qian et al. [228]	54.50	54.04
	Mohiuddin et al. [43]	60.00	59.93
	Ganguly et al. [44]	65.00	63.80
	Proposed	66.50	76.72
FF++_CeDF	Li et al. [175]	58.06	55.60
	Afchar et al. [42]	66.41	65.58
	Qian et al. [228]	63.89	53.89
	Mohiuddin et al. [43]	63.71	56.43
	Ganguly et al. [44]	68.04	66.12
	Zhao et al. [237]	-	67.44
	Miao et al. [238]	-	66.12
	Proposed	79.34	87.58

4.4.5 Comparison with Other FS Methods

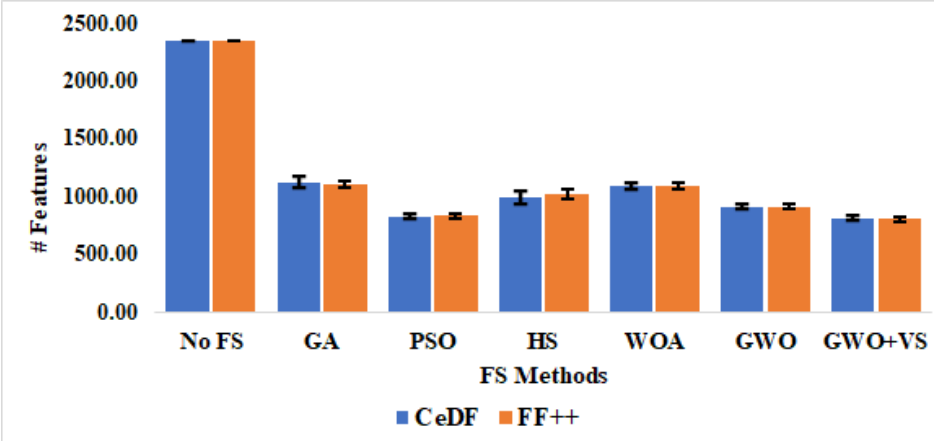
In this study, a hybrid FS technique, termed the GWO-VS method, is designed. Experiments are conducted to compare the performance of the proposed hybrid FS technique with other SOTA FS methods. For comparison, five widely used meta-heuristic techniques: Genetic Algorithm (GA), PSO, Harmony Search (HS), Whale Optimization Algorithm (WOA), and GWO are considered. To simplify the process, handcrafted features are concatenated with deep learning features before applying these FS techniques. Each experiment is repeated ten times, and the results are presented in Figure 4.16. The comparative results indicate that, in terms of average test accuracy and the number of selected features, the proposed hybrid FS technique outperforms the other methods. Furthermore, the superior performance of the GWO-VS based FS technique over the GWO-based FS method justifies the selection of GWO-VS within the proposed HFS framework.

4.4.6 Comparison with Past Deepfake Detection Methods

The performance of the proposed deepfake detection method is compared with several SOTA approaches [42, 107, 175, 228, 237, 238]. To ensure a fair evaluation and minimize biases introduced by different experimental setups, all methods, except [237, 238], have been evaluated on the current dataset splits (see subsection 4.2.2) and under the same experimental setup. Table 4.11 presents the comparative performance of all methods. The results indicate that the proposed method achieves performance comparable to existing SOTA techniques and outperforms the ViXNet model in all cases. Also, it



(a) Comparison in terms of test accuracy



(b) Comparison in terms of feature count

Figure 4.16: Comparison of accuracies for various SOTA FS methods on the CeDF and FF++ datasets. "No FS" refers to classification performed using the original features (i.e., Feature_5).

outperforms all other methods on the CeDF dataset, it does not achieve SOTA results on the FF++ dataset in terms of the AUC score. It is important to note that the experimental setup differs from the approaches proposed by Zhao et al. [237] and Miao et al. [238] in terms of the number of samples used for training and testing, the process of preparing cropped face images, and the number of frames per video. These differences may have contributed to the lower performance observed compared to these methods.

Table 4.11: Performance of the proposed model in the intra-dataset experimental setup.

Dataset	Methods	Model	# Features	Test Accuracy (in %)	AUC Score (in %)
CeDF	Li et al. [175]	Xception	2048	95.37	98.88
	Afchar et al. [42]	MesoNet	1024	65.64	65.33
	Afchar et al. [42]	MesoInception	1024	65.83	64.80
	Qian et al. [228]	F3-Net	4096	87.06	81.48
	Guo et al. [107]	AMTEN	300	68.33	78.04
	Methodology 1	ViXNet	-	94.40	98.54
	Methodology 2	HFS	270	**97.30**	**99.35**
FF++	Li et al. [175]	Xception	2048	96.00	98.34
	Afchar et al. [42]	MesoNet	1024	67.50	66.81
	Afchar et al. [42]	MesoInception	1024	65.00	66.93
	Qian et al. [228]	F3-Net	4096	95.50	96.52
	Guo et al. [107]	AMTEN	300	78.50	87.62
	Zhao et al. [237]	Multi-Attention + Xception	-	96.37	98.97
	Zhao et al. [237]	Multi-Attention + EfficientNet-B4	-	97.60	99.29
	Miao et al. [238]	FRLM	-	97.63	**99.50**
	Methodology 1	ViXNet	-	97.00	98.57
	Methodology 2	HFS	297	**98.00**	99.16

Table 4.12: Performance of the proposed HFS model and its components (see Figure 4.13) on DFDC dataset.

Parameter	Feature_H	Feature_D	Feature_1	Feature_2	Feature_3	Feature_4	Feature_5	Feature_6
Test Accuracy (in %)	49.91	69.19	51.91	72.03	66.83	75.34	71.81	73.07
AUC Score (in %)	51.01	77.59	53.66	79.88	75.63	85.67	80.19	81.24
# Features	300	2048	102	653	755	240	2348	859

4.4.7 Performance on the DFDC Dataset

The experiment on the DFDC dataset, previously described in subsection 4.2.8 with detailed experimental protocol, includes a comparison of the ViXNet model with several SOTA methods. In this section, the same results table is revisited, with the addition of the performance metrics for the proposed HFS-based method to facilitate a direct comparison with ViXNet and other SOTA models. The HFS-based method achieves an AUC score that is 0.65% lower than that of the previously reported ViXNet model on the DFDC dataset. The slight drop in AUC for the HFS-based method may be due to the higher complexity and diversity of the DFDC dataset. While HFS selects key features effectively, it may miss some modality-specific cues captured by ViXNet's end-to-end learning.

Table 4.13: Comparison of model performance on the DFDC dataset. Methods marked with * were evaluated under the experimental setup described in this work.

Method	#Training Samples	Performance (in %)	
		AUC Score	F1-Score
Heo et al. [230]	$> 1,000,000$	97.80	91.90
Wodajo et al. [35]	162,174	84.30	77.00
Guo et al. [107]*	86,166	74.19	72.83
Coccomini et al. [229]	220,444	92.50	84.50
Mohiuddin et al. [43] *	86,166	71.77	68.00
Ganguly et al. [44]*	86,166	83.59	74.21
Method 1	86,166	86.32	79.06
Method 2	86,166	85.67	75.34

4.5 Discussion

This chapter presents two effective methodologies designed to capture both local and global inconsistencies by extracting corresponding features for deepfake detection. The first proposed method, ViXNet, focuses on identifying forged facial regions by exploiting subtle, imperceptible residual artifacts left behind during the manipulation process. It utilizes a patch-wise self-attention module to generate face-specific masks and analyzes inconsistencies across these masked regions. These local inconsistencies are then combined with global spatial features extracted by a deep CNN, enabling robust classification. The second approach employs an HFS strategy that combines handcrafted and deep features extracted from the input data, where the PWSA path of ViXNet model is replaced by the handcrafted feature extraction process while the other path remains unchanged. Furthermore, it focuses on selecting only the most rele-

vant features from the combined set of handcrafted and deep features, while discarding redundant or non-contributory ones. This not only reduces the dimensionality of the feature space but also contributes to improved detection accuracy and computational efficiency. To assess the effectiveness of both methods, comprehensive experiments were carried out on three widely recognized deepfake detection datasets: FaceForensics++, Celeb-DF (V2), and DFDC. These datasets encompass a broad spectrum of real-world forgery scenarios, including challenging cases where the distinction between forged and genuine content is minimal due to low visual contrast. The experimental outcomes demonstrate the robustness and applicability of the proposed models in detecting deepfake under diverse and complex conditions.

Chapter 5

Inter-frame Video Forgery Detection

Digital media forgeries have predominantly focused on spatial alterations, where individual pixels or groups of pixels are manipulated to fabricate content. However, with the increasing prevalence of viral videos on social media platforms, temporal forgeries in videos have emerged as a significant concern. These forgeries, which alter the timing and sequence of frames, pose challenges not only for investigative agencies and journalists but also for ordinary users attempting to verify a video's authenticity before sharing. Tampered videos can influence political views, disrupt law and order, exacerbate international conflicts, and incite social unrest, making their detection critically important in today's digital age. Failure to verify video authenticity can lead to biased decisions, especially in contexts such as court proceedings where video evidence holds substantial weight.

Video forensics, the scientific discipline concerned with verifying the authenticity and integrity of video content, includes video tampering detection as a core subdomain. This area focuses on identifying content alterations and localizing tampered regions either spatially or temporally. Techniques in this domain can be classified as active or passive [3], depending on the availability of prior information about the media (as discussed in Chapter 2). Given that a video is essentially a sequence of frames over time, many image forensic methods [239] are applicable at the frame level.

Inter-frame video forgery is a specific type of temporal manipulation that disrupts the temporal consistency of a video by altering its frame sequence. The most common types are frame duplication, which repeats certain frames to extend or fabricate actions, and frame deletion, which removes frames to hide or distort events. These manipulations compromise the reliability of video content and pose substantial threats to digital security, legal integrity, and media authenticity. Detecting such forgeries necessitates advanced computational techniques, including statistical anomaly detection to identify irregularities in motion and frame dependencies.

Objectives of the Chapter

The key objectives covered in this chapter are as follows:

- Create statistical feature-based algorithms for detecting inter-frame forgeries, including frame duplication and deletion, without requiring prior training or

labeled data, and to ensure their effectiveness under various transformations such as brightness changes, noise, and blurring.

- Improve the precision of forgery detection by leveraging statistical analysis to identify manipulated frames and ensure accurate localization.
- Integrate derivative-based signal analysis to enhance sensitivity to subtle temporal anomalies introduced by frame-level manipulations.
- Capture temporal inconsistencies by transforming frame-wise differences into spatial representations for effective analysis.

To address the objectives outlined, two inter-frame forgery detection methodologies are introduced:

1. The first targets duplicate frame detection using sequence matching, where the SSIM is employed to assess the resemblance between consecutive frames. A dedicated search algorithm is then used to identify and localize the tampered segments.
2. The second method explores frame deletion detection through temporal inconsistency analysis. It begins by transforming frame-level distortions into spatial signals, which are then examined using a CNN to identify potential tampering. In practice, this involves computing MSE values between consecutive frames, visualizing these as curve plots, and using them as input for classification. Upon detecting manipulation, a more localized detection is performed using second-order derivative analysis to highlight abrupt shifts that may indicate deleted frames. The method's resilience is further tested under various video conditions, including alterations in noise, blur, and brightness.

5.1 Frame Duplication Detection Technique

SSIM is used to compare the similarity between two images by measuring changes in structural information, luminance, and contrast. It provides a similarity score that helps identify visually similar content. The proposed method detects duplicate frames in both surveillance (static background) and non-surveillance (non-static background) videos affected by copy-move forgery. Video copy-move refers to duplicating a segment of frames from one part of a video and inserting it elsewhere within the same video to conceal or falsify events. Since SSIM can compute a similarity score between consecutive video frames, applying it across an entire video generates a one-dimensional

vector referred to as the SSIM chain. This sequence effectively captures the temporal similarity structure of the video. In the presence of a copy-move attack, where a set of frames is duplicated and inserted elsewhere in the same video, the duplicated segments are likely to produce repeated patterns within the SSIM chain. Based on this observation, a method is proposed to detect frame duplication by identifying these recurring sequences. The key modules of this method are illustrated in Figure 5.1 and detailed in the following subsections.

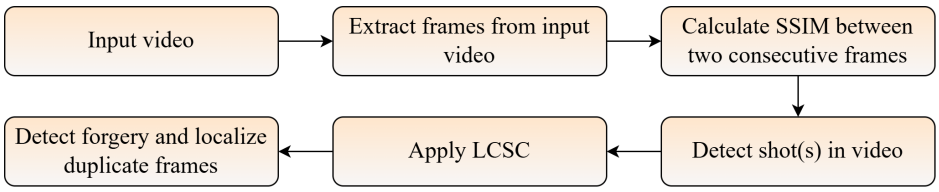


Figure 5.1: Key modules of the proposed method for detecting duplicate frames in manipulated videos.

5.1.1 Measuring Similarity between Two Frames

The SSIM is used to calculate the similarity between two frames by comparing their luminance, contrast, and structure. SSIM represents the perceptual difference between two frames but does not determine which one is of higher quality. It evaluates pixel-wise similarities by checking whether the intensity values of corresponding pixels in both frames are similar. The SSIM score ranges from -1 to 1, where 1 indicates high similarity, and -1 denotes significant differences. The steps to calculate the SSIM index are briefly discussed below.

SSIM [53] is widely used in image analysis to measure image quality or compare two images. Given a reference grayscale image R and a test image T, both of size $M \times N$, the SSIM between R and T is defined as:

$$SSIM\ (R,\ T)\ =\ l(R,\ T)\ *\ c(R,\ T)\ *\ s(R,\ T) \tag{5.1}$$

where

$$l(R,T) = \frac{2\mu_R\mu_T + C_1}{\mu_R^2 + \mu_T^2 + C_1},\quad c(R,T) = \frac{2\sigma_R\sigma_T + C_2}{\sigma_R^2 + \sigma_T^2 + C_2},\quad s(R,T) = \frac{\sigma_{RT} + C_3}{\sigma_R\sigma_T + C_3} \tag{5.2}$$

In Equation 5.1, the first term $l(R,T)$ represents luminance comparison, measuring the similarity in mean luminance between the two images. The second term $c(R,T)$ evaluates contrast differences, known as the contrast comparison function. The third term $s(R,T)$ measures the correlation coefficient between images R and T, referred

to as the structure comparison function. Constants C_1, C_2, and C_3 account for the saturation effect in low-brightness and contrast areas while ensuring numerical stability when the denominator approaches zero.

5.1.2 Detection of Shots in a Video

A video consists of multiple frames, denoted as $\{VF_1, VF_2, ..., VF_N\}$, where N represents the total number of frames. Before processing, all frames are converted to grayscale images. The SSIM is then used to assess the similarity between consecutive frames, as defined in Equation 5.1. The similarity between VF_i and VF_{i+1} is represented by S_i and is calculated as:

$$S_i = SSIM(VF_i, VF_{i+1}) \tag{5.3}$$

where $i = 1, 2, \ldots, N-1$.

Frames are segmented into multiple parts whenever the similarity index S_i falls below a predefined threshold τ. Each segment is referred to as a shot. According to Zheng et al. [240], the human eye and brain cannot effectively process frame differences occurring at a rate of less than 10 frames per second. To prevent false shot detection, a constraint is applied, ensuring that each shot has a minimum length of 10 frames. If a shot is shorter than this threshold, it is merged with the previous shot. The threshold value τ is computed as:

$$\tau = \mu_S - \sigma_S \tag{5.4}$$

where μ_S and σ_S represent the mean and standard deviation of all S_i values, respectively, calculated as:

$$\mu = \frac{1}{N} \sum_{i=1}^{N} s_i \tag{5.5}$$

$$\sigma = \sqrt{\frac{1}{N} \sum_{i=1}^{N} |s_i - \mu|^2} \tag{5.6}$$

After segmenting the frame sequence, a set of shots $\{SHOT_1, SHOT_2, ..., SHOT_P\}$ is obtained, where each shot has a variable length $\{L_1, L_2, ..., L_P\}$ and $P < N$, with P representing the total number of detected shots.

5.1.3 Frame Duplication Detection

After obtaining multiple shots of varying lengths in the previous step, the next task is to identify any frame duplication between different shots. This is achieved by searching

for the largest common identical frame sequence using the LCSC technique, as outlined in Algorithm algorithm 5.1. The similarity between frames from different shots is determined by calculating the SSIM score, as defined in Equation 5.3. However, directly computing SSIM scores for every possible frame pair across two shots of lengths L_1 and L_2 would require $L_1 \times L_2$ function calls, significantly increasing computational cost. To optimize this process, previously computed SSIM scores from consecutive frames in the entire video, as described in subsection 5.1.2, are utilized. Let S_{mi} and S_{nj} represent the SSIM scores of the i^{th} frame of $SHOT_m$ and the j^{th} frame of $SHOT_n$, respectively. Two frames are considered identical if $|S_{mi} - S_{nj}| < M_C$, where M_C is a threshold value determined experimentally. Algorithm 5.1 returns the indices of duplicated frames. When comparing two shots, multiple common sequences may be detected. Any sequence with a length below 10 is discarded. The complete method is summarized in Algorithm 5.2.

Algorithm 5.1: Pseudo code for LCSC search method

Input: $SHOT_m[1..p]$, $SHOT_n[1..q]$
Output: All indices of the matched sequence
$A \leftarrow$ array of size (p, q);
$L \leftarrow 0$ `// Holds the length of the longest common sequence`
for $x \leftarrow 1$ ***to*** p **do**
  for $y \leftarrow 1$ ***to*** q **do**
    if $SHOT_m[x]$ *is identical to* $SHOT_n[y]$ **then**
      if $x = 1$ ***or*** $y = 1$ **then**
        $A[x, y] \leftarrow 1$;
      else
        $A[x, y] \leftarrow A[x - 1, y - 1] + 1$;
      end
      if $A[x, y] > L$ **then**
        $L \leftarrow A[x, y]$;
        Store start and end indices of the first matched sequence from both SHOTs;
      end
      else if $A[x, y] = L$ **then**
        Further detected sequence of the same length as the first one, storing new indices along with the previous;
      end
    end
    else
      $A[x, y] \leftarrow 0$;
    end
  end
end
return *All indices of the matched sequence*;

Algorithm 5.2: Algorithm for Frame Duplication Detection

Input: N: Number of video frames
Output: Indices of detected copied sequences
$C \leftarrow$ Minimum number of frames a shot must contain and be duplicated in a forged video;
for $i \leftarrow 2$ ***to*** N **do**
  $S_i \leftarrow \text{SSIM}(VF_{i-1}, VF_i)$;
end
$\tau \leftarrow \mu_S - \sigma_S$;
//Frame sub-sequence generation;
$Count \leftarrow 1$;
$k \leftarrow 1$;
for $i \leftarrow 1$ ***to*** $N - 1$ **do**
  if $S_i < \tau$ ***and*** $Count > C$ **then**
    Break at position i;
    $SUB_k \leftarrow$ Store all SSIM values from the previous break point;
    $Count \leftarrow 1$;
    $k \leftarrow k + 1$;
  end
  else
    $Count \leftarrow Count + 1$;
  end
end
//Matching of sub-sequences;
for $i \leftarrow 1$ ***to*** $length(SHOT) - 1$ **do**
  for $j \leftarrow i + 1$ ***to*** $length(SHOT)$ **do**
    $matched \leftarrow \text{LCSC}(SHOT_i, SHOT_j)$;
    if $length(matched) \geq C$ **then**
      return *Indices of detected copied sequences*;
    end
  end
end

5.2 Experiment Results and Analysis

The proposed model is evaluated using a custom and challenging dataset carefully designed to simulate real-world forgery scenarios. Details of the dataset and the corresponding evaluation results are provided in the following subsections.

5.2.1 Dataset Design for Frame Duplication Attack Detection

We have compiled our dataset by selecting original videos from the DERF collection[1], REWIND[2], Urban Tracker [241], and YouTube. The dataset includes 19 videos with a

[1] `https://media.xiph.org/video/derf/`
[2] `https://sites.google.com/site/rewindpolimi/downloads/datasets`

static background and 18 videos with a non-static background, encompassing a variety of formats such as uncompressed, AVI and compressed MP4. The video resolutions range from a maximum of 626×352 pixels to a minimum of 320×240 pixels. All videos have been compressed using the libx264 and libavcodec libraries in FFmpeg[3].

The dataset consists of:

- **10 videos** without frame duplication.
- **10 videos** with single-instance frame duplication.
- **3 videos** with multiple-instance frame duplication.

Additionally, 14 modified videos have been included to evaluate the robustness of our method in noisy conditions. These consist of 9 videos with one-time frame duplication and 5 videos without frame duplication, altered through operations such as sharpness adjustment, vintage effects, blurring, and noise addition. In total, our dataset comprises 37 videos, each with a duration of approximately 6 to 18 seconds, and a frame rate varying between 25 and 30 fps. A summary of the forged videos used in our experiments is presented in Table 5.1.

5.2.2 Evaluation Metrics

The performance of the proposed system is assessed using three statistical measures: detection accuracy (DA) (see Equation 3.7), true positive rate (TPR) or Recall (see Equation 3.9), and true negative rate (TNR). The TNR is defined as

$$TNR = \frac{TN}{TN + FN} \tag{5.7}$$

TPR and TNR, also known as sensitivity and specificity, respectively, provide insights into system effectiveness. TPR measures the ability to correctly identify forged frames, while TNR evaluates the likelihood of correctly classifying real frames. Here, TP represents the number of correctly predicted forged frames, TN denotes the correctly predicted real frames, FP corresponds to incorrectly predicted forged frames, and FN represents incorrectly predicted real frames.

5.2.3 Results and Discussion

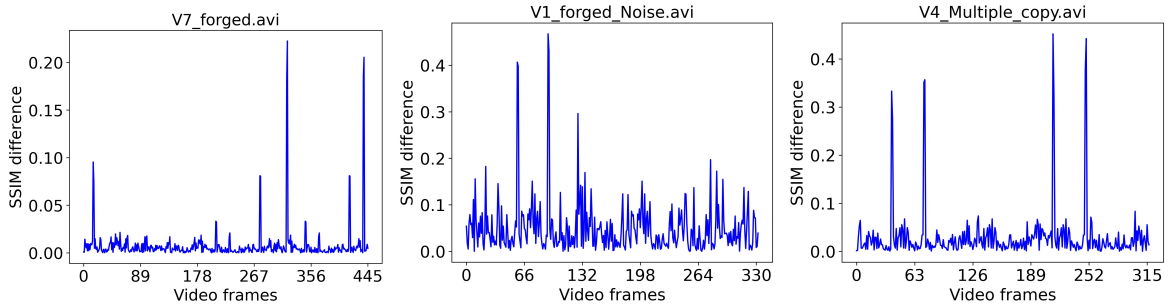
As discussed in subsection 5.1.3, the decision on whether two frames are identical is based on a threshold value (M_C). To determine this threshold, the absolute differences between two SSIM scores in a video were analyzed. It was observed that, in the absence of frame duplication, this absolute difference does not exceed 0.300. Consequently,

[3] `https://ffmpeg.org/`

Table 5.1: The current dataset includes forged videos with frame duplication. The first 10 videos contain single-instance frame duplication, while the last three exhibit multiple-instance frame duplication.

Video	**Original Length**	**Tampered Length**	**Forgery Operation**
V1_forged.avi	300	335	185:220 copied at 60:95
V2_forged.avi	300	430	0:130 copied at 170:299
V3_forged.avi	235	335	220:320 copied at 0:100
V4_forged.avi	250	350	60:160 copied at 180:280
V5_forged.avi	180	300	170:290 copied at 20:140
V6_forged.avi	360	470	10:120 copied at 320:430
V7_forged.avi	321	447	180:300 copied at 320:440
V8_forged.avi	209	286	10:87 copied at 200:277
V9_forged.avi	317	417	316:416 copied at 80:180
V10_forged.avi	130	250	5:125 copied at 130:250
V1_Multiple_copy.avi	300	350	10:35 copied at 50:75 and 100:125
V2_Multiple_copy.avi	300	350	155:180 copied at 70:95 and 210:235
V4_Multiple_copy.avi	250	320	135:170 copied at 40:75 and 215:250

this value was selected as the threshold for the current experimentation. Figure 5.2 illustrates three cases where the consecutive SSIM score differences remain below M_C, except when frame duplication occurs.



(a) **(b)** **(c)**

Figure 5.2: An illustration of SSIM score differences, where a significant increase is observed at the copied positions, while remaining below M_C in other cases: (a) Frames 180–300 copied to 320–440, (b) Frames 185–220 copied to 60–95, and (c) Frames 135–170 copied to 40–75 and 215–250.

The videos are grouped into four major categories to facilitate a better analysis of the system's performance. Category 1 includes videos with no forgery, Category 2

contains videos with a single frame duplication forgery, Category 3 consists of videos with post-processing operations (both original and forged), and Category 4 contains videos with multiple frame duplication forgeries. Table 5.2 presents the category-wise average DA, TPR, and TNR, along with the overall average performance achieved by the proposed method.

Table 5.2: Presents both category-wise and overall performance of the proposed frame duplication detection technique.

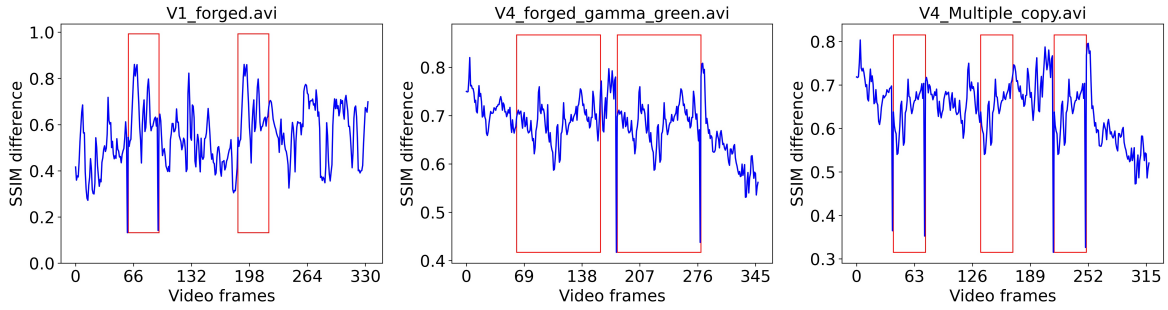
	#videos	**DA**	**TPR**	**TNR**
Category 1	10	0.989	1.0	0.989
Category 2	10	0.985	1.0	0.978
Category 3	14	0.990	1.0	0.988
Category 4	03	1.0	1.0	1.000
Average	37	0.989	1.0	0.987

Analysis of the results in Table 5.2 reveals that the TPR values for all video categories are 1.0, indicating that the system effectively detects forged frames. However, while identifying forged frames, some real frames are misclassified as forged, leading to lower TNR values in the first, second, and third video categories. In contrast, the system performs best on videos with multiple frame duplication forgeries. Figure 5.3 illustrates three examples of detected duplicate frames, with one example selected from each of the last three video categories. Additionally, the system's performance is evaluated on videos with both static and non-static backgrounds. The proposed method maintains satisfactory performance in these cases as well (refer to Table 5.3). When tested on videos with non-static backgrounds, the method detects both forged and original frames with 100% accuracy. However, some errors occur in videos with static backgrounds, where real frames from shots with no object movement are mistakenly identified as forged.

Table 5.3: Duplicate frame detection results for the videos with static and non-static background

Type of video	**#videos**	**DA**	**TPR**	**TNR**
Static background videos	19	0.979	1.0	0.974
Non-static background videos	18	1.0	1.0	1.0

Additionally, the performance of the proposed method was tested on videos (both original and frame duplication videos) affected by various distortion operations, as mentioned earlier. The method performs well in these cases as well. Table 5.4 presents the detection performance on post-processed videos.



(a) **(b)** **(c)**

Figure 5.3: Localization of frame duplication for (a) Category 2, (b) Category 3, and (c) Category 4 videos.

Table 5.4: Presents the frame duplication detection performance on videos affected by distortion operations such as noise addition, vintage effects, and contrast changes.

	#videos	**DA**	**TPR**	**TNR**
Category 3 (Original videos with post-processing)	5	0.989	1.0	0.989
Category 3 (One time frame duplication videos with post-processing)	9	0.991	1.0	0.987
Average	14	0.990	1.0	0.988

5.3 Frame Deletion Forgery Detection

A video shot is a continuous sequence of frames captured by a single camera without interruption, typically showing an unbroken segment of action from a fixed angle. Removing multiple frames from a shot often causes sudden object shifts, making the tampering visually detectable. However, deleting an entire shot from surveillance footage where the camera is stationary and consecutive frames are similar can go unnoticed by both humans and automated systems. This makes detecting such deletions particularly challenging in surveillance videos with static backgrounds and fixed cameras [242].

Verifying the authenticity of frame deletion forgery in videos raises two critical questions, particularly in the context of legal evidence or other sensitive domains:

- **Is the video authentic?** This question aims to determine whether the video has undergone frame deletion forgery.
- **Where did the frame deletion occur?** If tampering is detected, this step focuses on localizing the exact position of the deleted frames within the forged video.

Researchers have explored two main approaches for detecting frame removal in videos: analyzing temporal inconsistencies between consecutive frames [243, 244] and using deep learning to classify videos as authentic or tampered [245, 246]. Some methods detect only frame deletion [193, 247], while a few handle multiple inter-frame forgeries like insertion, deletion, and duplication [99, 248]. However, most focus on locating tampering rather than verifying its presence. Deep learning methods face challenges due to the lack of real-world forged data and often rely on synthetic deletions [245, 249], which limits generalization. They also require extensive resources and training time, reducing practicality for real-time use [250].

To address these limitations, a two-stage technique is proposed that answers both key questions by integrating a CNN based frame deletion forgery detection method with an efficient second order derivative aided approach for locating deleted frames. An overview of the detection framework is provided in Figure 5.4.

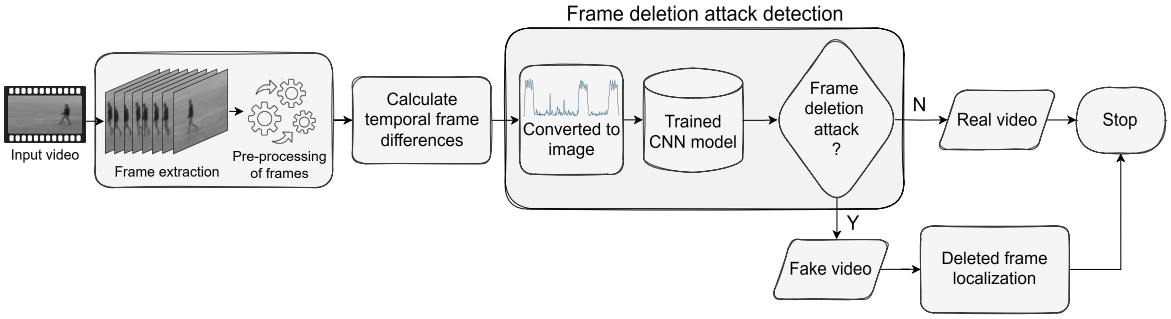


Figure 5.4: Overview of the video frame deletion detection methodology.

5.3.1 Preprocessing

The video is first decomposed into individual frames. Each frame is a 24-bit color image comprising red (R), green (G), and blue (B) channels. These frames are converted into grayscale using Equation 5.8.

$$F = 0.299 \times R + 0.587 \times G + 0.114 \times B \tag{5.8}$$

In Equation 5.8, F denotes the grayscale intensity value derived from the RGB components.

5.3.2 Detection of Frame Deletion Attack

Following preprocessing, consecutive frames are compared using metrics such as MSE, PSNR, SSIM, and Normalized Cross-Correlation (NCC) to evaluate inter-frame differences in the temporal domain. In this method, MSE is employed to quantify the difference between consecutive frames. MSE calculates the average squared difference

in pixel intensity values between two images. The MSE value, denoted as m_t, for two consecutive frames F_t and F_{t+1} (where $t \in \{1, 2, \ldots, N-1\}$ and N is the total number of frames in the video), each of size $H \times W$, is computed using Equation 5.9.

$$m_t = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} \left(F_t(i,j) - F_{t+1}(i,j)\right)^2 \tag{5.9}$$

In Equation 5.9, $F_t(i,j)$ and $F_{t+1}(i,j)$ represent the pixel intensity values at position (i,j) in the t^{th} and $(t+1)^{th}$ frames, respectively. The resulting sequence of MSE values is then transformed into a visual representation by plotting it as a graph, creating an image-based form (MSE plot) that is used as input for the deep learning model. Figure 5.5 shows sample MSE plots for three original videos (a, b, c) and their corresponding forged versions (d, e, f), illustrating the variations introduced by frame deletion. CNNs are effective at extracting spatial patterns. This property is leveraged by using MSE plots without axis labels (e.g., frame numbers and MSE values), enabling the model to learn temporal inconsistencies through spatial cues.

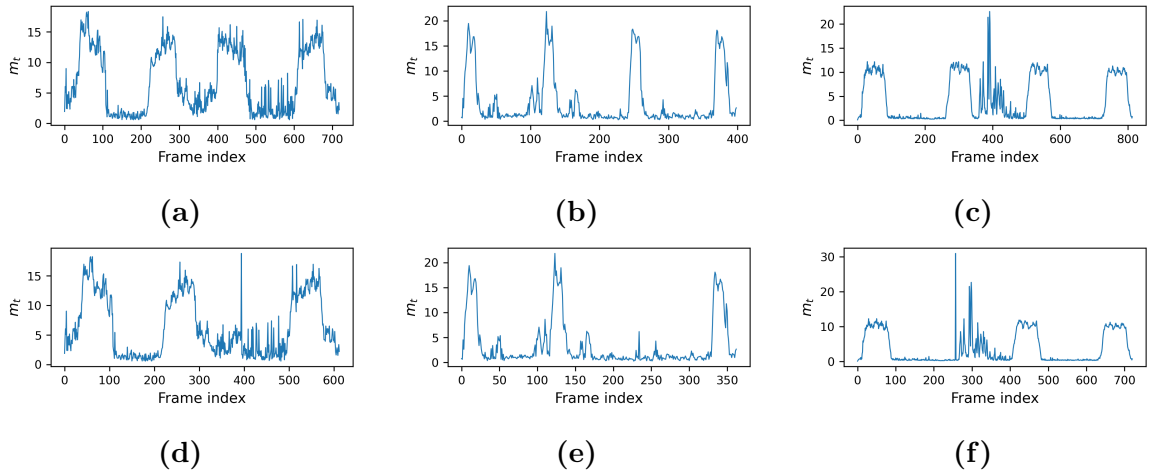


Figure 5.5: Examples of MSE plots for real (top row) and manipulated (bottom row) videos. Frame deletion artifacts are reflected as inconsistencies in the curve shapes.

To classify these plots, a lightweight CNN model is introduced as the backbone architecture. Its low complexity supports deployment on resource-constrained platforms [251–253] while aligning with sustainable AI objectives [254–257]. The architecture is based on MesoNet [42], with modifications incorporating depthwise separable convolutions to reduce the number of trainable parameters and computational requirements. The adapted CNN model contains approximately 0.094 million trainable parameters, offering a 31% reduction compared to the original MesoNet model, which has 0.137 million parameters.

The proposed model accepts input images of size $112 \times 336 \times 3$ and consists of

four sequential convolutional blocks. Each block includes a `SeparableConv2D` layer followed by batch normalization, a ReLU activation function, and a max pooling layer. The first two convolutional layers employ 3×3 kernels with 16 filters, while the third and fourth layers utilize 32 filters. Pooling layers use a stride of 2 and apply same padding to progressively reduce spatial resolution. After feature extraction, the output is flattened and passed through two dense layers with 64 and 16 neurons, respectively, both activated by ReLU and regularized with a dropout rate of 0.3. A final sigmoid-activated neuron performs binary classification, distinguishing between real and tampered inputs.

The overall model architecture is depicted in Figure 5.6, and the stepwise process for detecting frame deletion is outlined in Algorithm 5.3.

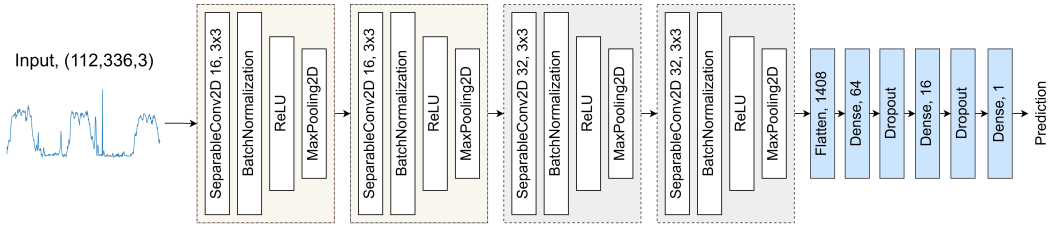


Figure 5.6: CNN architecture used in Algorithm 5.3 for detecting frame deletion in videos based on input MSE plots.

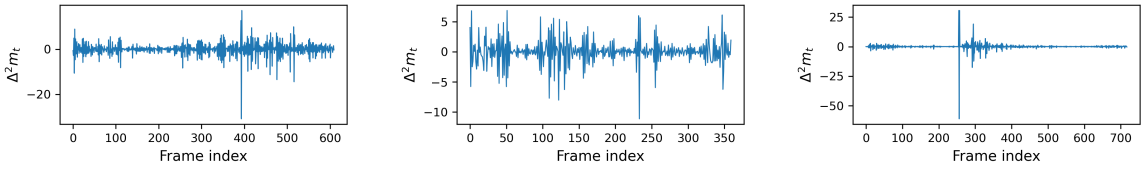
5.3.3 Localization of Deleted Frames

Upon detecting a frame deletion attack within a video, the subsequent objective is to determine the exact temporal locations from which the frames have been removed. This process is referred to as "localization of the deleted frames". To accomplish this, the second-order derivative of the inter-frame MSE values is utilized. The methodology and implementation steps involved in this process are detailed in this section.

The inter-frame difference curve generally exhibits a gradual increase when an object enters a static scene and a gradual decrease when the object exits. Such slow fluctuations are not useful as visual indicators for precise localization. In contrast, abrupt changes in the curve are more informative, as illustrated in Figure 5.5d and Figure 5.5e, where sudden spikes and flat regions occur despite scene changes. To enhance sensitivity to abrupt changes and suppress gradual trends, the second-order derivative is computed over the inter-frame difference values. This derivative highlights significant fluctuations that may correspond to tampered segments.

The second-order derivative of m_t (for $t \in \{1, 2, \ldots, N-1\}$) is calculated to capture the rate of change in the difference values. This measure, also referred to as curvature or acceleration, is defined in Equation 5.10:

$$\Delta^2 m_t = m_{t+1} - 2 * m_t + m_{t-1} \tag{5.10}$$



(a) **(b)** **(c)**

Figure 5.7: Second-order derivative curves of MSE values corresponding to the three previously analyzed manipulated videos, highlighting fluctuations attributed to frame deletion.

In Equation 5.10, $\Delta^2 m_t$ denotes the second-order derivative of m_t. A high positive value suggests a sudden increase in inter-frame variation, whereas a high negative value indicates a sudden drop. Figure 5.7 presents the second-order derivative curves for the three manipulated videos shown earlier in Figure 5.5 (d, e, f), where sharp changes correspond to potential frame deletions. To minimize false detections, a threshold τ is computed by combining the mean (μ) and standard deviation (σ) of the second-order derivative values with an empirical scaling factor α, as shown in Equation 5.11:

$$\tau = \mu + \alpha * \sigma \tag{5.11}$$

The initial detection point set DP is initialized as empty (Equation 5.12), and candidate frame indices t satisfying the condition in Equation 5.13 are added to DP.

$$DP = \phi \tag{5.12}$$

$$DP = DP \cup \{t\}, \text{ if } |\Delta^2 m_t| > \tau \tag{5.13}$$

Algorithm 5.4 provides a structured step-by-step description of this localization process.

5.4 Experimental Evaluation and Observations

A robust forensic detection framework must effectively distinguish between authentic and tampered videos across varying datasets and manipulation types. In this study, a two-phase detection strategy is adopted. The first phase determines the presence of frame deletion in the input video. Upon detection, the second phase estimates the precise temporal positions of the deleted frames. The subsequent sections present

Algorithm 5.3: Detection of Frame Deletion Attack

Input: Video V with a static background
Output: Authenticity flag $flag$ indicating whether the video is tampered
Extract all frames F_t from video V;
Convert each frame to grayscale using Equation 5.8;
for *each consecutive frame pair* (F_t, F_{t+1}) **do**
  Compute m_t using Equation 5.9;
end
Plot the m_t values as a curve image;
Feed the curve image into the CNN model for classification;
if *CNN output indicates Fake* **then**
  $flag \leftarrow 0$;
  `// Video is classified as tampered`
  Execute Algorithm 5.4;
else
  $flag \leftarrow 1$;
  `// Video is classified as authentic`
end
return $flag$;

information regarding the dataset, experimental results, in-depth discussions, and an analysis of observed failure cases.

5.4.1 Dataset Description

To the best of current knowledge, no publicly available standard dataset exists for evaluating newly developed video frame deletion detection techniques. In existing literature, researchers typically construct their own datasets to perform experiments. Following this practice, a custom dataset was created using sample videos from [258], commonly referred to as the "Recognition of Human Actions" database. This dataset includes six categories of human actions: walking, jogging, running, boxing, hand waving, and hand clapping each performed multiple times by 25 subjects under 4 different conditions: outdoors, outdoors with scale variation, outdoors with different clothing, and indoors.

The constructed dataset contains 600 videos, with 100 videos allocated to each action category. All videos were recorded using a static camera at a frame rate of 25 frames per second, featuring homogeneous backgrounds. The spatial resolution of each video is 160×120 pixels, and the average duration is approximately 4 seconds. From the full dataset, only the categories of walking, jogging, and running are selected, resulting in a subset of 300 original (real) videos. These categories are chosen due to their consistent scenes and lack of scale variation such as zooming. Annotations indicating the start and end frame indices of individual shots segments in which a

Algorithm 5.4: Localization of Deleted Frames

Input: MSE values m_t for $t \in \{1, 2, \ldots, N-1\}$ from Algorithm 5.3
Output: Detected deletion points DP
for *each* $t \in \{1, 2, \ldots, N-1\}$ **do**
  Compute $\Delta^2 m_t$ using Equation 5.10;
end
Compute threshold τ using Equation 5.11;
Initialize $DP \leftarrow \emptyset$;
for *each* $t \in \{1, 2, \ldots, N-1\}$ **do**
  if $|\Delta^2 m_t| > \tau$ **then**
    $DP \leftarrow DP \cup \{t\}$;
  else
    continue;
  end
end
return DP;

person appears and performs an action are provided.

To create manipulated (fake) videos, one or two consecutive shots were removed programmatically from each original video, excluding the initial and final shots. This approach ensures that the starting and ending points of each video remain unchanged. As a result, 300 fake videos with deleted frames were generated, yielding a total of 600 videos in the dataset. In contrast, the remaining action categories consist of a single continuous shot; therefore, frame removal from these would cause noticeable discontinuities, making manipulation easily perceptible. The selected categories consist of multiple shots, enabling more seamless frame deletion that is less detectable by visual inspection. The code used to perform frame deletion has been made publicly available[4]. However, the modified videos cannot be shared due to licensing restrictions imposed by the dataset owner [258]. Figure 5.8 illustrates examples of the video sequences with highlighted shots.

5.4.2 Evaluation Metrics

To evaluate classification performance, four standard metrics are used: Accuracy (Equation 3.7), Precision (Equation 3.8), Recall (Equation 3.9), and F1 score (Equation 3.10). These metrics provide a comprehensive assessment of the model's ability to distinguish between real and fake videos.

[4] `https://github.com/sk-mohiuddin/Frame-Deletion-Detection`

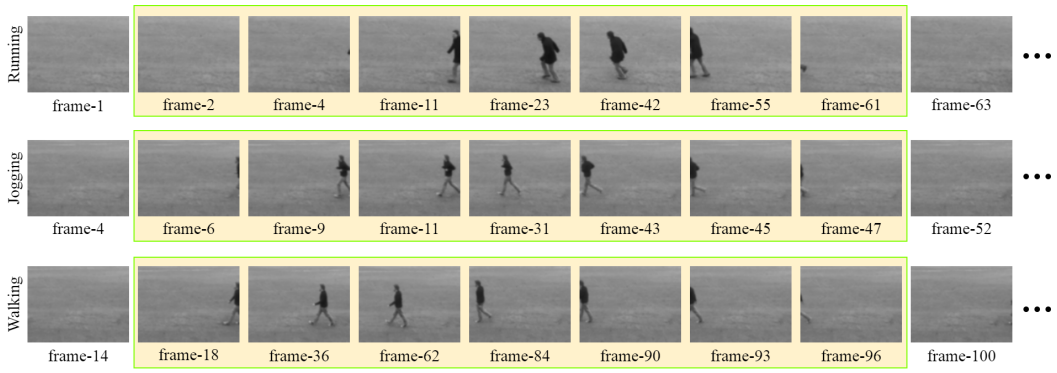


Figure 5.8: Example frame sequences for the actions: running, jogging, and walking. Each yellow-highlighted region denotes a distinct shot, and the corresponding frame indices are shown below each frame. The sequences include multiple shots. When consecutive shots are removed, the adjacent frames before and after the removed segment become continuous. For example, in the running sequence, frames 1 and 63 appear consecutively; in jogging, frames 4 and 52; and in walking, frames 14 and 100. This results in similar-looking frames appearing next to each other.

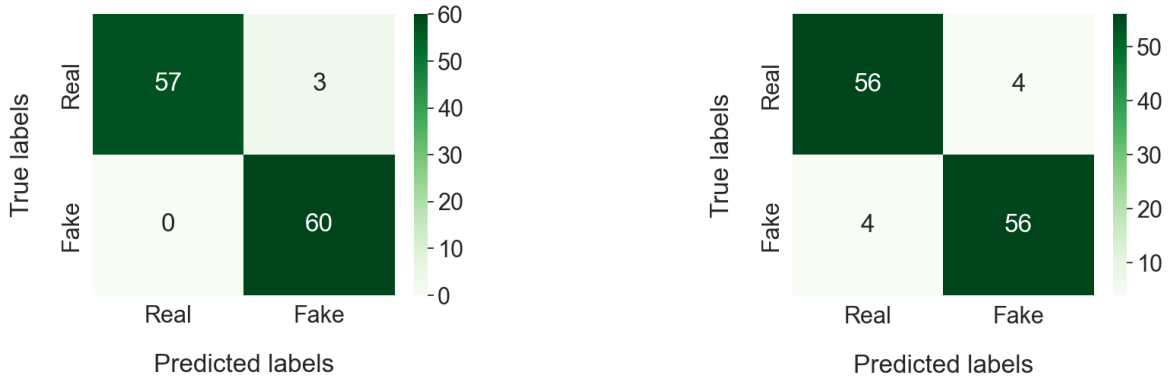
5.4.3 Experimental Setup

The proposed model was implemented in Python using TensorFlow and Keras. Supporting libraries including NumPy, OpenCV, and scikit-learn were utilized for preprocessing and data manipulation. All input images, specifically the MSE curve plots, were resized to a standard dimension of 112×336 pixels. A batch size of 16 was used to optimize memory usage and training efficiency. The Adam optimizer was employed with a learning rate of 0.001 to ensure stable convergence. A consistent train-test split of 80:20 was used across all experiments. Additionally, five-fold cross-validation was applied to assess the generalization of the model on the MSE curve plots. For all experiments, the threshold of α in Equation 5.11 was set to 4.0.

5.4.4 Performance of Frame Deletion Attack Detection

To determine whether a video has been subjected to a frame deletion attack, a lightweight CNN-based classification approach was utilized (refer to Algorithm 5.3). Five-fold cross-validation was performed on the curated 600-video dataset to evaluate detection performance. Each fold consisted of 120 samples with equal representation of real and fake videos, and no overlap between folds. Results are summarized in Table 5.5. The model achieved an average accuracy of 96.00% with a standard deviation of 1.71%, indicating consistent classification performance across folds. Fold-1 produced the lowest accuracy (93.33%), while other folds demonstrated higher values. These outcomes validate the reliability and efficiency of the lightweight CNN architecture for early stage forgery detection, with potential for deployment in near real-time

applications. Confusion matrices for the best and worst performing folds are displayed in Figure 5.9.



(a) Highest accuracy case (Fold-3). **(b)** Lowest accuracy case (Fold-1).

Figure 5.9: Confusion matrices for the best and worst results from five-fold cross-validation. (a) represents Fold-3 with the highest accuracy and F1 score, while (b) shows Fold-1 with comparatively lower performance.

Table 5.5: Performance results of frame deletion detection using standard evaluation metrics. Results are presented as mean (x) $\pm$ standard deviation (y).

Fold	Accuracy	Precision	Recall	F1 Score
Fold-1	93.33%	93.33%	93.33%	93.33%
Fold-2	97.50%	98.31%	96.67%	97.48%
Fold-3	97.50%	95.24%	100.00%	97.56%
Fold-4	95.83%	92.31%	100.00%	96.00%
Fold-5	95.83%	93.65%	98.33%	95.94%
Average	96.00 $\pm$ 1.53	94.57 $\pm$ 2.09	97.67 $\pm$ 2.50	96.06 $\pm$ 1.53

5.4.5 Performance of Deleted Frames Localization

The performance of frame deletion localization was evaluated using second-order computations based on frame differences, as outlined in Algorithm 5.4. This experiment was conducted on all 300 fake videos, each containing deleted segments. The localization technique accurately detected the deletion positions in 298 out of 300 videos, yielding a localization accuracy of 99.33%. The application of the second-order derivative of the frame difference curve, denoted as $\Delta^2 m_t$, played a significant role in enhancing localization. It effectively reduced false positives by suppressing minor fluctuations that do not correspond to actual deletions, thereby enabling precise identification of the deleted segments. This result demonstrates the robustness and effectiveness of the localization algorithm.

5.4.6 Performance of the Proposed Method

Sections 5.4.4 and 5.4.5 presented the performance of individual modules in the proposed two-stage approach, namely the forgery detection component described in Algorithm 5.3 and the frame deletion localization component detailed in Algorithm 5.4. This section provides an evaluation of the complete two-stage framework.

A holdout test sample strategy was employed, where 60 real and 60 fake videos (a total of 120 videos) are randomly selected from the complete dataset for testing. The remaining videos were used for training the CNN-based classifier used in the forgery detection module. The classifier achieved an overall accuracy of 97.50%, correctly classifying 117 out of 120 test samples. The confusion matrix corresponding to this classification is shown in Figure 5.10. According to the matrix, one fake video was misclassified as real and two real videos were misclassified as fake, resulting in a false positive rate of 3.33% and a false negative rate of 1.67%.

Subsequently, the 61 videos identified as fake were passed to the frame deletion localization module. Among these, 58 videos were correctly localized, yielding a localization accuracy of 95.08%. It is important to note that the localization module identified incorrect deletion points for both of the real videos erroneously labeled as fake in the previous stage. Excluding these two false positives and considering only the actual fake videos (59 videos), the localization accuracy increased to 98.30%. These results confirm that the proposed approach is not only effective in detecting frame deletion forgeries but also accurate in pinpointing the manipulated segments.

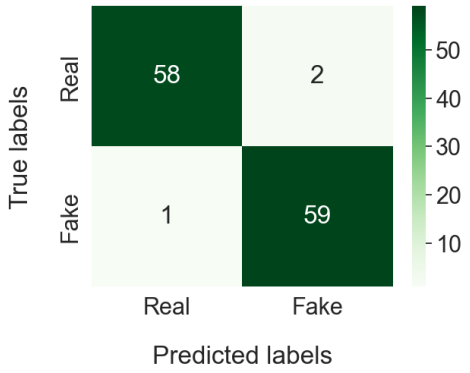


Figure 5.10: Confusion matrix representing the classification performance of the CNN-based forgery detection module using the test set described in Section 5.4.6.

5.4.7 Performance on Post-processed Videos

To assess the robustness of the proposed two-stage frame deletion detection and localization framework, post-processing operations were applied to all 600 videos (real and fake) in the dataset prior to evaluation using Algorithm 5.3. The objective was to

examine the system's performance under various visual distortions commonly encountered in real-world scenarios. The applied modifications included brightness increase (BI), brightness decrease (BD), Gaussian blurring (GB), and the addition of Gaussian noise (GN). The parameters used for these transformations are listed in Table 5.6. A 5-fold cross-validation protocol was adopted, with each fold containing 120 post-processed videos (60 real and 60 fake), ensuring no video overlap between folds. The performance of the detection algorithm under each condition is reported in Table 5.6.

Without any post-processing, the frame deletion attack detection method yields an average accuracy of 96.00% and an average F1-score of 96.06%. The BI operation lead to a decrease in accuracy by 1.89% and a reduction in F1-score by 2.06%. Under BD, the accuracy has declined by 1.38% and the F1-score by 1.23%. In contrast, the GB condition had no impact on accuracy and a minimal F1-score reduction of 0.01%, indicating high resilience to blurring. The GN operation has resulted in a 0.86% drop in accuracy and a 0.77% drop in F1-score. These results confirm that the proposed method exhibits strong robustness, especially under Gaussian blur, with only slight degradations under brightness and noise alterations. The variations in frame difference curves due to post-processing are shown in Figure 5.12, based on the three previously examined fake videos from Figure 5.5.

To further analyze the impact on the localization module (see Algorithm 5.4), all 300 post-processed fake videos are evaluated. The baseline performance without post-processing achieved the highest localization accuracy, correctly identifying the deleted frame locations in 298 out of 300 videos (99.33%). After applying BI, the accuracy decreased by 2.67% (290 correctly localized videos). For GB and BD, 293 videos are accurately localized, each reflecting a 1.67% reduction. The GN scenario has shown relatively better robustness with 296 correct localizations, indicating a 0.67% decrease. Among all the conditions, BI caused the highest degradation. The corresponding results are visualized in Figure 5.11.

5.4.8 Analysis of Failure Cases

This section presents an analysis of the failure cases, with Figure 5.13 providing graphical illustrations of the frame difference curves and their corresponding second-order derivatives for videos that are misclassified. The first row in Figure 5.13 (a, b, c) displays the frame difference curves of three fake videos that are incorrectly classified by the CNN model described in Algorithm 5.3 during the initial stage of frame deletion forgery detection, as discussed in Section 5.4.4. These videos exhibit fluctuation patterns that deviate from those typically observed in fake samples. One of the curves shows an absence of fluctuation at the frame deletion position. In these

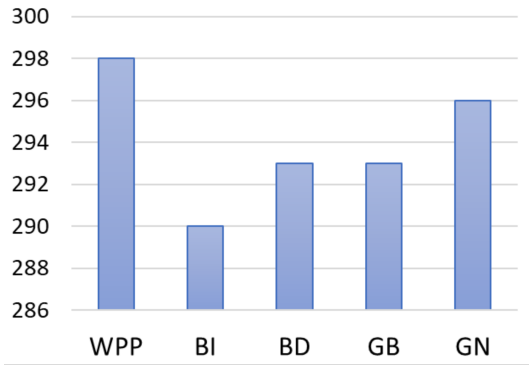


Figure 5.11: Bar chart showing the number of correctly localized deleted frames under various post-processing operations (BI, GB, BD, and GN). WPP represents the baseline scenario without post-processing, evaluated using Algorithm 5.4.

Table 5.6: Detection performance of the proposed method under different post-processing conditions. The evaluation metrics are presented as mean ± standard deviation across 5-fold cross-validation. BI, BD, GB, and GN refer to brightness increase ($\alpha = 1.10$, $\beta = 20$, Factor = +10%), brightness decrease ($\alpha = 0.90$, $\beta = -20$, Factor = -10%) , Gaussian blurring (Kernel $(5, 5)$, $\sigma = 1.0$), and Gaussian noise ($\sigma = 4$), respectively.

Post-processing	Accuracy	Precision	Recall	F1-Score
BI	94.17 ± 1.86	95.33 ± 4.82	93.33 ± 6.45	94.08 ± 2.06
BD	94.67 ± 2.25	92.44 ± 5.16	97.67 ± 1.90	94.88 ± 1.99
GB	96.00 ± 0.91	94.26 ± 1.26	98.00 ± 2.17	96.07 ± 0.93
GN	95.17 ± 1.24	92.92 ± 3.68	98.00 ± 2.17	95.32 ± 1.07

cases, the presence of false peaks or the lack of expected peaks at the deleted frames in the difference curves could contribute to incorrect classification, as the model fails to effectively distinguish between authentic and manipulated content.

In the second row, all plots correspond to a single fake video. Figure 5.13 (d) shows the original frame difference curve, while Figure 5.13 (e) illustrates the same curve after the introduction of noise. Figure 5.13 (f) presents the second-order derivative of the noisy sequence. Although Algorithm 5.3 accurately identifies the video as manipulated, the localization of the deleted frame is unsuccessful due to noise. The added noise disturbs the natural pattern of the curve, leading to multiple threshold crossings at different positions and thereby reducing the accuracy of localization, as discussed in Section 5.4.7. This example demonstrates the sensitivity of the proposed technique to postprocessing artifacts and emphasizes the influence of noise on detection precision.

The third row, comprising Figure 5.13 (g, h), illustrates the impact of camera instability. The video contains several pronounced peaks induced by unsteady motion, resulting in irregular curve variations. These peaks exceed the predefined threshold

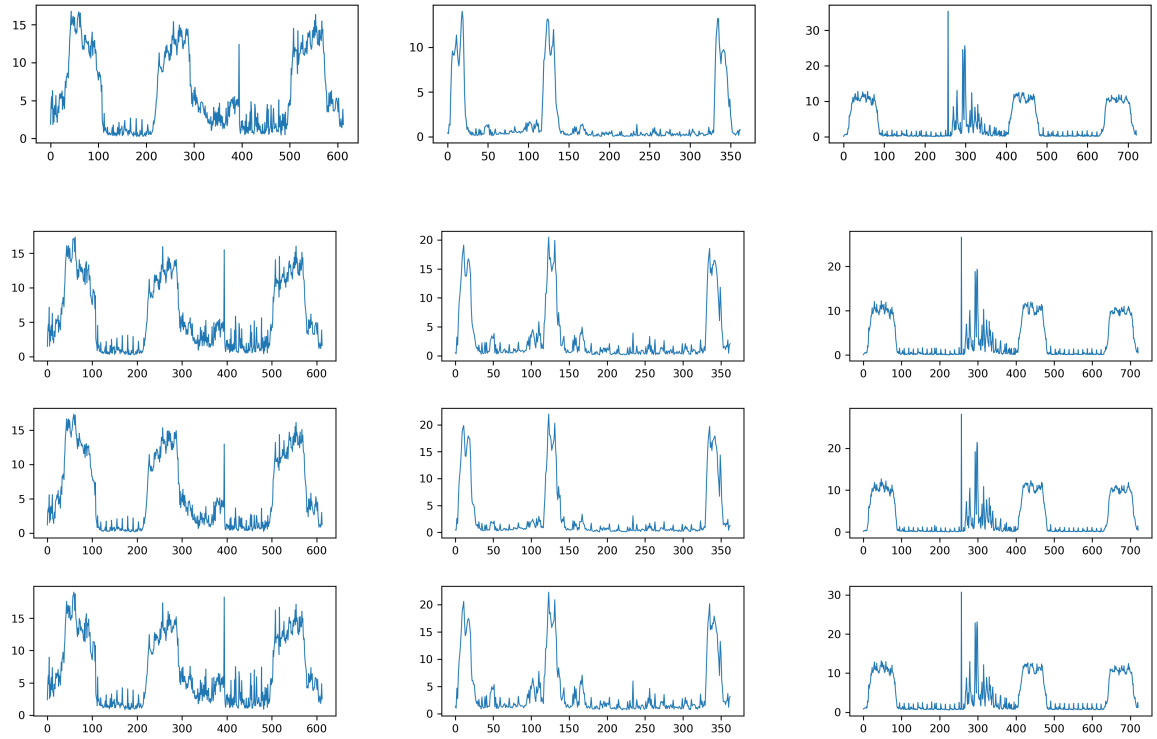


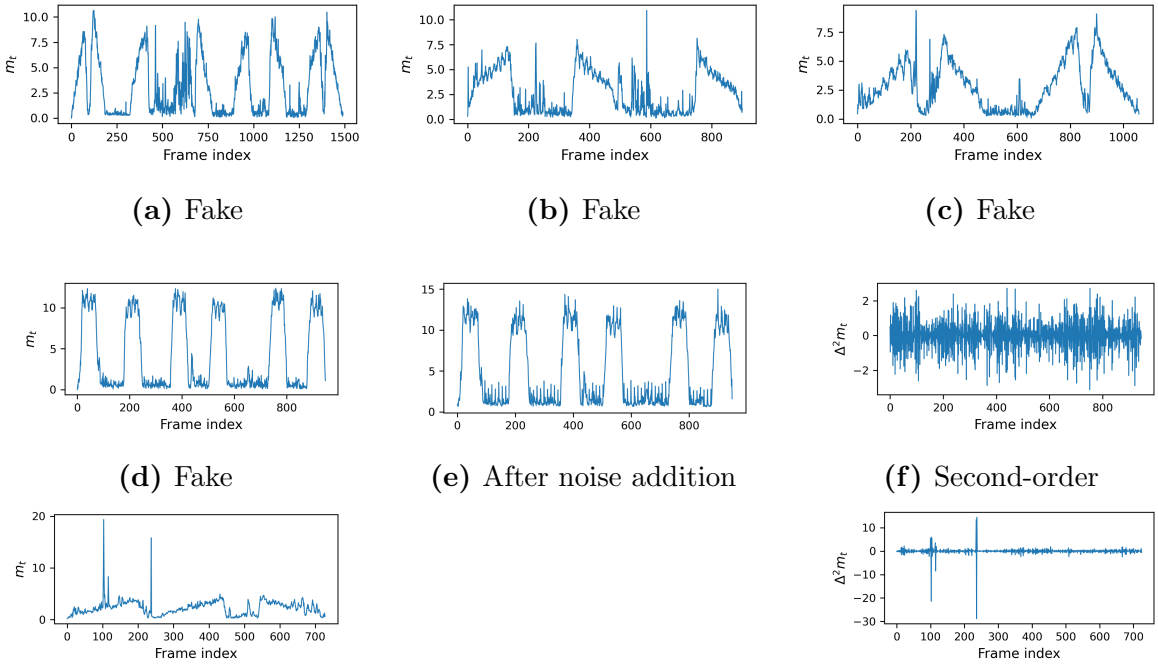
Figure 5.12: Visual comparison of MSE curve variations following different post-processing operations applied to the same set of fake videos depicted in Figure 5.5. Rows correspond to: (a) brightness increase, (b) brightness decrease, (c) Gaussian blur, and (d) Gaussian noise.

τ, which leads to incorrect localization, as described in subsection 5.4.5. The overall analysis indicates that factors such as camera stability, motion characteristics, and image quality degradation significantly influence the behavior of MSE curves. The proposed method effectively captures these variations and is capable of identifying frame deletions and distortions in diverse conditions, although its accuracy may be affected in the presence of noise and unstable motion.

5.5 Discussion

This chapter is presented inter-frame forgery detection methods, each addressing a distinct type of tampering. To detect frame duplication forgery, the proposed method employs the SSIM to evaluate the resemblance between consecutive frames. A dedicated searching algorithm is then applied to identify and localize the duplicated frames. For the experimental evaluation, original videos are sourced from the Urban Tracker, DERF collection, and REWIND databases, and forged videos were generated through the duplication of selected frames.

For detecting frame deletion forgery, a two-stage method is introduced through a



(a) Fake **(b)** Fake **(c)** Fake

(d) Fake **(e)** After noise addition **(f)** Second-order

(g) Fake **(h)** Second-order

Figure 5.13: Failure analysis of the proposed method based on frame difference and second-order derivative curves. Various conditions such as noise, motion instability, and abnormal patterns are shown to affect detection and localization performance.

structured verification pipeline. The input video first undergoes preprocessing, during which inter-frame differences between consecutive frames are calculated. These differences are converted into an image format, transforming temporal inconsistencies into spatial representations. A CNN is used to analyze this data and classify whether frame deletion has occurred or not. Upon detection of tampering, the method proceeds to localize the deleted frames by applying second-order derivative analysis to the inter-frame difference data. The proposed method is evaluated under various video conditions, including alterations in noise, blur, and brightness levels, to assess its robustness.

Chapter 6

Conclusion and Future Scope

The rapid advancement of digital editing tools, AI, and deep learning technologies has significantly simplified the process of altering multimedia content. These innovations enable the seamless modification of images and video, often making manipulated content indistinguishable from authentic material. While such technologies have benefited industries like entertainment, education, and healthcare, they have also opened the door to malicious misuse. Forged image/video content can be used to spread misinformation, impersonate individuals, or damage reputations. As these techniques become more sophisticated, detecting and mitigating manipulated media has become a critical area of research in digital forensics.

This thesis has focused on exploring existing approaches to forgery detection in images and videos, particularly those presented as evidence in investigative contexts. The discussion began with a comprehensive literature review, which played a crucial role in identifying key research gaps and formulating the primary objectives of the present study. Through this review, it became evident that while a range of techniques exists spanning from traditional statistical methods to advanced deep learning frameworks, many suffer from limitations in generalization, robustness across diverse conditions, temporal coherence in video analysis, and localization accuracy of manipulated regions. The study critically examined various methodologies, including pixel-level analysis, frequency domain techniques, ML classifiers, and modern neural network-based solutions. Special attention is given to hybrid models that combine deep features with handcrafted features, as they offer a balanced trade-off between accuracy and efficiency. Additionally, the literature highlighted a growing interest in attention-based mechanisms and multi-scale architectures, particularly in deepfake detection tasks. This detailed survey of prior work has laid a strong foundation for the research by not only mapping the evolution of digital forgery detection methods but also by revealing specific challenges such as handling low-quality or heavily compressed media, detecting subtle manipulations, and ensuring robustness across diverse datasets. Consequently, the study has been structured into several components, each aimed at addressing a particular category of forgery, including image copy-move, deepfake content and inter-frame video forgeries. The insights gained from this critical review directly inform the design and development of the proposed methods, ensuring

that the research is both grounded in existing knowledge and oriented toward solving current limitations in the field.

To tackle the challenges of image CMFD discussed in **Chapter 3**, this study introduces an improved and resource-efficient MultiResUNet model, specifically tailored for this task. The proposed model significantly reduces complexity, containing only 3.26 million parameters compared to the original 7.27 million. The architecture incorporates key improvements to increase detection accuracy and robustness while maintaining computational efficiency. A core component of the model is the MultiRes blocks, which utilizes varied kernel sizes to extract multi-scale features. This multi-resolution strategy allows the model to effectively capture both fine-grained and large-scale forgeries. Additionally, the integration of Residual blocks enhances the flow of gradients and supports efficient information exchange between the encoder and decoder layers, thus promoting stable and deeper network training. A key contribution of this work is the introduction of a patch-wise similarity-based positional attention module SPAM. This module operates by calculating cosine similarity between image patches to highlight spatial inconsistencies, enabling precise localization of duplicated regions. By incorporating this attention mechanism, the model becomes more sensitive to subtle manipulations that may be overlooked by traditional convolutional filters alone. The proposed method is rigorously evaluated on four widely used and challenging CMFD benchmark datasets: CoMoFoD, COVERAGE, CASIA TIDE v2.0, and MICC-F600. These datasets include a diverse range of tampering scenarios, such as varying illumination, post-processing effects, and low-contrast forgeries. The proposed method achieves strong performance across multiple benchmark datasets while utilizing less than half the parameters of the original MultiResUNet model. Notably, it attained an F1-score of 84.51 and Accuracy of 99.34% on the CoMoFoD dataset, 79.81 / 98.57% on Coverage, and 93.96 / 96.28% on MICC. Even on the more challenging CASIA v2 dataset, the model achieved a competitive F1-score of 74.97%. Furthermore, the results confirm the superiority of deep learning-based techniques over conventional keypoint-based and block-matching methods, particularly in detecting complex and high-resolution forgeries. The proposed model not only ensures accurate identification of manipulated regions but also shows strong resistance to false matches and spurious detections, which are common pitfalls in traditional approaches. Overall, the findings validate the effectiveness of the MultiResUNet-based system. Its ability to operate efficiently while maintaining high detection performance makes it a practical solution for real-world image forensics applications, especially in scenarios requiring precise and reliable tampering detection.

With the proliferation of synthetic media, deepfake (presents in **Chapter 4**) have

emerged as a significant threat to reputation, democratic processes, and public perception. These AI-generated face images can convincingly mimic real individuals, making it increasingly difficult to distinguish between authentic and manipulated content. The primary motivation is to extract both local and global features from face images to identify manipulation artifacts. While existing methods often focus on limited components of the facial region, this approach aims to detect artifacts across the entire face, ensuring a more comprehensive analysis of potential tampering. To address deepfake detection challenges in images and videos, various methods are used to explore manipulation artifacts by focusing on both global and local features. A holistic strategy is adopted in the early stage of this study, where global features are extracted from entire face images using the Xception model, and local inconsistencies are captured through face patch segmentation. A soft attention mechanism is applied to emphasize regions with potential artifacts, enhancing the model's focus on manipulated areas. Additionally, the impact of different color models—RGB, HSV, and YCbCr is evaluated by processing images through separate MesoInception4 networks, and the extracted features are averaged to improve detection accuracy. It achieves 85.93% acc and 85.92% AUC on FF++, and 84.56% acc with 78.46% AUC on CeDF, highlighting its effectiveness across diverse datasets. These early-stage methods highlight the significance of color representation and attention-based feature enhancement in boosting deepfake detection performance.

Subsequently, to address more advanced challenges, this research develops an improved method called ViXNet, which combines local and global feature extraction strategies to effectively detect face-swap deepfake. It begins by segmenting each facial image into patches and applying a patch-wise self-attention mechanism that emphasizes subtle artifacts present in manipulated regions. This local attention process includes a masking operation, where each patch is multiplied by a trainable weight matrix to enhance manipulation-specific features. The masked patches are then processed through a ViT which captures global inconsistencies by modeling the relationships between patches across the entire image. In parallel, ViXNet incorporates the Xception network as a global feature extractor to capture broader spatial patterns and textures often disrupted in deepfake. Finally, the local features from the transformer and global features from the Xception model are stacked together and passed through a deep fully connected classifier. This classifier, composed of multiple dense layers followed by a softmax activation, predicts the probability of an image being real or fake. The integrated architecture allows ViXNet to detect subtle manipulations by jointly analyzing both intra-patch and inter-patch inconsistencies along with global facial structure, making it robust against various face-swapping techniques. ViXNet

was evaluated using both intra- and inter-dataset experiments on three benchmark datasets: FaceForensics++, Celeb-DF (V2), and DFID. It achieved AUC scores of 98.57% (83.60%), 99.26% (74.78%), and 98.93% (75.13%) for intra(inter)-dataset settings, respectively. On the DFDC dataset, ViXNet attained an AUC of 86.32% and an F1-score of 79.06%. The model demonstrating strong robustness and generalization capability across different datasets.

In addition, to emphasize local manipulation artifacts, another deepfake detection method adopts a composite approach that integrates both global and local features. Deep feature-based methods often generate numerous features, many of which may be redundant or irrelevant, increasing computational cost and acting as a black box. These challenges emphasize the need for effective feature selection to remove unnecessary features. This approach explores the effectiveness of combining deep and handcrafted features through a hierarchical feature selection method that aims to improve detection performance while reducing redundancy. Deep features are extracted from face images using the pre-trained XceptionNet model, known for its ability to capture fine-grained spatial patterns. Simultaneously, handcrafted features are obtained by dividing each cropped face into 25 patches of size 45 by 45 pixels and extracting Daisy descriptors from each patch with the scikit-image library, resulting in a 2600-dimensional vector. Euclidean distances between all patch pairs are calculated to capture internal inconsistencies, forming a 300-dimensional feature vector. These deep and handcrafted features are concatenated into a unified vector and then refined using hierarchical feature selection based on the GWO-VS method. Finally, classification into real or fake categories is performed using a Support Vector Machine with an RBF kernel, chosen for its effectiveness in handling high-dimensional data. The model achieves AUC scores of 99.35% on Celeb-DF (V2), 99.16% on FaceForensics++, and 85.67% on DFDC, using only 11.50%, 12.65%, and 10.22% of the original features, respectively. Additionally, both the model outperforms most SOTA methods identified in the literature and evaluated on these three datasets.

Further, to address the challenges of inter-frame video forgery detection (presents in **Chapter 5**), specifically in the temporal domain for identifying frame duplication, a method based on statistical features has been proposed. Such manipulations can be critical in real-world scenarios, as they may be used to present fake surveillance footage with the intent to mislead investigations or influence court decisions. This approach integrates SSIM-based shot segmentation with the LCSC algorithm. The process begins by converting the video into individual frames and segmenting them into distinct shots using SSIM, which ensures accurate detection of shot boundaries. Within each segmented shot, the LCSC algorithm is applied to identify duplicated frames

with high accuracy. This two-stage strategy improves the robustness and effectiveness of the detection process while maintaining computational efficiency. For experiments, original videos from Urban Tracker, DERF, and REWIND datasets were used to create forged samples via frame duplication. The method achieved an average detection accuracy of 98.90% on these videos. In addition, this study introduces another method focused on detecting frame deletion forgery in surveillance videos. This technique relies on analyzing the MSE differences between successive frames. The video is first preprocessed to compute MSE values between consecutive frames, which are then used to create a curve plot representing temporal inconsistencies in a spatial format. This plot is input to a modified MesoNet CNN model that classifies whether the video has experienced frame deletion. If tampering is detected, the method proceeds to identify the exact points of deletion. A second-order derivative analysis is applied to the MSE values to detect sharp changes, which indicate potential frame removals. To ensure the reliability of this method, additional experiments are conducted by modifying core video characteristics such as noise, blur, and brightness. The proposed method achieves an average accuracy of 96.00% in detecting frame deletion and 98.30% in localizing the deleted frames, evaluated on a custom dataset.

Limitations of the Thesis

The methods employed in this study encounter some challenges, which are outlined below:

For the proposed method in image CMFD, while it effectively identifies duplicated regions, the detection around boundary areas remains imprecise. This leads to performance degradation, particularly when forgeries involve complex textures or irregular patterns. It is essential not only to detect the presence of duplicated regions but also to distinguish which segment is the original and which is the forged copy. This distinction is crucial, as the duplicated region can obscure important background details, potentially hiding key information or altering the visual context. The current model is unable to distinctly identify the source and destination regions of the copied segments, which limits the forensic value of the detection results.

The deepfake detection methods proposed in this study have certain limitations, including the imbalance in dataset samples where real and fake instances are not equally represented, which affects model training. Furthermore, models trained on specific datasets often struggle to generalize to real-world or previously unseen forgeries. While the patch-wise self-attention mechanism improves artifact localization in case of ViXNet, it increases model complexity and computational load, and like most transformer-based architectures, it requires large-scale data to generalize effectively.

Although spatial features are well captured, the models struggle to exploit temporal inconsistencies such as unnatural blinking or motion artifacts, limiting their performance on advanced video deepfake. The full pipeline—feature extraction, selection, and classification is computationally intensive, making real-time or low-resource deployment challenging. Moreover, handcrafted features like Daisy or Haralick are sensitive to noise and geometric distortions, reducing robustness across different resolutions and compression levels.

For the proposed inter-frame video forgery detection techniques, the use of preset threshold dependent metrics limits the ability to detect subtle and localized manipulations. Moreover, applying these methods to long video sequences in real-time scenarios remains computationally intensive, reducing their practicality in real-world forensic applications. Additionally, the limited availability of publicly accessible datasets, which often lack diversity and do not reflect real-life scenarios, further constrains the evaluation and generalization of the proposed methods.

Directions for Future Research

Although the techniques presented in this thesis yield promising results compared to current SOTA methods, there remains scope for further enhancement. Potential directions for future work include the following:

1. In image CMFD, while duplicated regions are effectively identified, the model struggles with precise detection at region boundaries, particularly in the presence of complex textures or irregular patterns. Future work may focus on refining boundary localization of forged region.

2. Generalization remains a critical issue, as models trained on specific datasets often underperform on previously unseen or real-world manipulations. Cross-dataset training and evaluation, domain adaptation techniques, and adversarial training may be employed to address this challenge.

3. Deepfake detection methods primarily focus on spatial artifact detection. However, they fall short in capturing temporal inconsistencies, such as unnatural blinking or motion artifacts, which are common in video deepfake. Integrating spatio-temporal models like 3D CNNs or temporal attention networks could improve detection accuracy in video-based forgeries.

4. For inter-frame video forgery detection, the reliance on fixed threshold values limits the system's ability to detect subtle and localized manipulations. Adaptive thresholding mechanisms, possibly informed by ML, could be introduced to enhance flexibility and detection accuracy.

5. Lastly, the limited availability and diversity of publicly accessible video forgery datasets hinder comprehensive evaluation. Future efforts should focus on curating and publishing large-scale, diverse, and realistic datasets that reflect a wide range of manipulation techniques, video resolutions, and lighting conditions to support more robust training and benchmarking.

In summary, the work presented in this thesis is the outcome of extensive research and systematic experimentation aimed at addressing critical challenges in image and video forgery detection. The proposed methods address various forgery types, including image copy-move manipulations, deepfake, frame duplication and frame deletion by combining handcrafted, deep and statistical feature-based approaches. The use of advanced feature selection and classification techniques leads to more accurate and reliable detection outcomes. This unified approach enhances both the performance and forensic value of the system. The findings and resources from this work lay the groundwork for building practical, explainable, and scalable solutions for detecting digital forgeries in both images and videos.

References

[1] A. Rocha, W. Scheirer, T. Boult, and S. Goldenstein, "Vision of the unseen: Current trends and challenges in digital image and video forensics," *ACM Computing Surveys (CSUR)*, vol. 43, no. 4, pp. 1–42, 2011.

[2] S. Tyagi and D. Yadav, "A detailed analysis of image and video forgery detection techniques," *The Visual Computer*, vol. 39, no. 3, pp. 813–833, 2023.

[3] K. Sitara and B. M. Mehtre, "Digital video tampering detection: An overview of passive techniques," *Digital Investigation*, vol. 18, pp. 8–22, 2016.

[4] S. T. Nabi, M. Kumar, P. Singh, N. Aggarwal, and K. Kumar, "A comprehensive survey of image and video forgery techniques: variants, challenges, and future directions," *Multimedia Systems*, vol. 28, no. 3, pp. 939–992, 2022.

[5] G. Kaur, N. Singh, and M. Kumar, "Image forgery techniques: a review," *Artificial Intelligence Review*, vol. 56, no. 2, pp. 1577–1625, 2023.

[6] S. Mohiuddin, S. Malakar, M. Kumar, and R. Sarkar, "A comprehensive survey on state-of-the-art video forgery detection techniques," *Multimedia Tools and Applications*, vol. 82, no. 22, pp. 33499–33539, 2023.

[7] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," *Advances in neural information processing systems*, vol. 27, 2014.

[8] S. Suwajanakorn, S. M. Seitz, and I. Kemelmacher-Shlizerman, "Synthesizing obama: learning lip sync from audio," *ACM Transactions on Graphics (ToG)*, vol. 36, no. 4, pp. 1–13, 2017.

[9] BBC News, "Deepfake videos are becoming harder to detect," 2018. Accessed: 2025-06-03.

[10] N. Kaur, N. Jindal, and K. Singh, "Passive image forgery detection techniques: A review, challenges, and future directions," *Wireless Personal Communications*, vol. 134, no. 3, pp. 1491–1529, 2024.

[11] C. Serbanescu, "Why does artificial intelligence challenge democracy? a critical analysis of the nature of the challenges posed by ai-enabled manipulation,"

A Critical Analysis of the Nature of the Challenges Posed by AI-Enabled Manipulation (august 4, 2021). Serbanescu, C.," Why Does Artificial Intelligence Challenge Democracy, pp. 105–128, 2021.

[12] C. Vaccari and A. Chadwick, "Deepfakes and disinformation: Exploring the impact of synthetic political video on deception, uncertainty, and trust in news," *Social media+ society*, vol. 6, no. 1, p. 2056305120903408, 2020.

[13] H. Farid, "Exposing digital forgeries in scientific images," in *Proceedings of the 8th workshop on Multimedia and security*, pp. 29–36, 2006.

[14] J. Fridrich, "Digital image forensics," *IEEE Signal Processing Magazine*, vol. 26, no. 2, pp. 26–37, 2009.

[15] S. Agarwal and H. Farid, "Photo forensics from jpeg dimples," in *2017 IEEE workshop on information forensics and security (WIFS)*, pp. 1–6, IEEE, 2017.

[16] I. Amerini, L. Ballan, R. Caldelli, A. Del Bimbo, and G. Serra, "A sift-based forensic method for copy–move attack detection and transformation recovery," *IEEE transactions on information forensics and security*, vol. 6, no. 3, pp. 1099–1110, 2011.

[17] F. Marra, D. Gragnaniello, D. Cozzolino, and L. Verdoliva, "Detection of gan-generated fake images over social networks," in *2018 IEEE conference on multimedia information processing and retrieval (MIPR)*, pp. 384–389, IEEE, 2018.

[18] J. Kietzmann, J. Paschen, and E. Treen, "Artificial intelligence in advertising: How marketers can leverage artificial intelligence along the consumer journey," *Journal of Advertising Research*, vol. 58, no. 3, pp. 263–267, 2018.

[19] T. T. Nguyen, Q. V. H. Nguyen, D. T. Nguyen, D. T. Nguyen, T. Huynh-The, S. Nahavandi, T. T. Nguyen, Q.-V. Pham, and C. M. Nguyen, "Deep learning for deepfakes creation and detection: A survey," *Computer Vision and Image Understanding*, vol. 223, p. 103525, 2022.

[20] BBC Culture, "How a 19th-century portrait of abraham lincoln was later revealed to be a fake," Mar. 2024. Accessed: 2025-04-11.

[21] Wikipedia contributors, "Censorship of images in the soviet union." `https://en.wikipedia.org/wiki/Censorship_of_images_in_the_Soviet_Union`, 2024. Accessed: 2025-04-11.

[22] M. A. Qureshi and M. Deriche, "A review on copy move image forgery detection techniques," in *2014 IEEE 11th International Multi-Conference on Systems, Signals & Devices (SSD14)*, pp. 1–5, IEEE, 2014.

[23] G. K. Birajdar and V. H. Mankar, "Digital image forgery detection using passive techniques: A survey," *Digital investigation*, vol. 10, no. 3, pp. 226–245, 2013.

[24] P. Yin and H. H. Yu, "Classification of video tampering methods and countermeasures using digital watermarking," in *Multimedia Systems and Applications IV*, vol. 4518, pp. 239–246, SPIE, 2001.

[25] J. Bakas, A. K. Bashaboina, and R. Naskar, "Mpeg double compression based intra-frame video forgery detection using cnn," in *2018 International Conference on Information Technology (ICIT)*, pp. 221–226, IEEE, 2018.

[26] G.-Y. Kang, Y.-P. Feng, R.-K. Wang, and Z.-M. Lu, "Edge and feature points based video intra-frame passive-blind copy-paste forgery detection," *Journal of Network Intelligence*, vol. 6, no. 3, pp. 637–645, 2021.

[27] E. Aparicio-Díaz, R. Cumplido, M. L. Pérez Gort, and C. Feregrino-Uribe, "Temporal copy-move forgery detection and localization using block correlation matrix," *Journal of Intelligent & Fuzzy Systems*, vol. 36, no. 5, pp. 5023–5035, 2019.

[28] G. Chittapur, S. Murali, and B. S. Anami, "Video forgery detection using motion extractor by referring block matching algorithm," *Int J Sci Technol Res*, vol. 8, no. 10, pp. 3240–3243, 2019.

[29] C.-S. Lin and J.-J. Tsay, "A passive approach for effective detection and localization of region-level video forgery with spatio-temporal coherence analysis," *Digital Investigation*, vol. 11, no. 2, pp. 120–140, 2014.

[30] K. Kono, T. Yoshida, S. Ohshiro, and N. Babaguchi, "Passive video forgery detection considering spatio-temporal consistency," in *Proceedings of the Tenth International Conference on Soft Computing and Pattern Recognition (SoCPaR 2018) 10*, pp. 381–391, Springer, 2020.

[31] G. Qadir, S. Yahaya, and A. T. Ho, "Surrey university library for forensic analysis (sulfa) of video content," in *IET conference on image processing (IPR 2012)*, p. 121, IET, 2012.

[32] R. D. Singh and N. Aggarwal, "Detection of upscale-crop and splicing for digital video authentication," *Digital Investigation*, vol. 21, pp. 31–52, 2017.

[33] A. Malik, M. Kuribayashi, S. M. Abdullahi, and A. N. Khan, "Deepfake detection for human face images and videos: A survey," *Ieee Access*, vol. 10, pp. 18757–18775, 2022.

[34] C. Lu, B. Liu, W. Zhou, Q. Chu, and N. Yu, "Deepfake video detection using 3d-attentional inception convolutional neural network," in *2021 IEEE International conference on image processing (ICIP)*, pp. 3572–3576, IEEE, 2021.

[35] D. Wodajo and S. Atnafu, "Deepfake video detection using convolutional vision transformer," *arXiv preprint arXiv:2102.11126*, 2021.

[36] K. B. Meena and V. Tyagi, "Image forgery detection: survey and future directions," in *Data, Engineering and Applications: Volume 2*, pp. 163–194, Springer, 2019.

[37] M. Verma and D. Singh, "Survey on image copy-move forgery detection," *Multimedia Tools and Applications*, vol. 83, no. 8, pp. 23761–23797, 2024.

[38] N. B. Abd Warif, A. W. A. Wahab, M. Y. I. Idris, R. Ramli, R. Salleh, S. Shamshirband, and K.-K. R. Choo, "Copy-move forgery detection: survey, challenges and future directions," *Journal of Network and Computer Applications*, vol. 75, pp. 259–278, 2016.

[39] A. Heidari, N. Jafari Navimipour, H. Dag, and M. Unal, "Deepfake detection using deep learning methods: A systematic and comprehensive review," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 14, no. 2, p. e1520, 2024.

[40] N. A. Shelke and S. S. Kasana, "A comprehensive survey on passive techniques for digital video forgery detection," *Multimedia Tools and Applications*, vol. 80, no. 4, pp. 6247–6310, 2021.

[41] N. Ibtehaz and M. S. Rahman, "Multiresunet: Rethinking the u-net architecture for multimodal biomedical image segmentation," *Neural networks*, vol. 121, pp. 74–87, 2020.

[42] D. Afchar, V. Nozick, J. Yamagishi, and I. Echizen, "Mesonet: a compact facial video forgery detection network," in *2018 IEEE International Workshop on Information Forensics and Security (WIFS)*, pp. 1–7, IEEE, 2018.

[43] S. Mohiuddin, S. Ganguly, S. Malakar, D. Kaplun, and R. Sarkar, "A feature fusion based deep learning model for deepfake video detection," in *International Conference on Mathematics and its Applications in new Computer Systems*, pp. 197–206, Springer, 2021.

[44] S. Ganguly, S. Mohiuddin, S. Malakar, E. Cuevas, and R. Sarkar, "Visual attention-based deepfake video forgery detection," *Pattern Analysis and Applications*, vol. 25, no. 4, pp. 981–992, 2022.

[45] S. Ganguly, A. Ganguly, S. Mohiuddin, S. Malakar, and R. Sarkar, "ViXNet: Vision Transformer with Xception Network for deepfakes based video and image forgery detection," *Expert Systems with Applications*, vol. 210, p. 118423, 2022.

[46] S. Mohiuddin, K. H. Sheikh, S. Malakar, J. D. Velásquez, and R. Sarkar, "A hierarchical feature selection strategy for deepfake video detection," *Neural Computing and Applications*, vol. 35, no. 13, pp. 9363–9380, 2023.

[47] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1251–1258, 2017.

[48] G. Naskar, S. Mohiuddin, S. Malakar, E. Cuevas, and R. Sarkar, "Deepfake detection using deep feature stacking and meta-learning," *Heliyon*, vol. 10, no. 4, 2024.

[49] S. Mohiuddin, A. Roy, S. Pani, S. Malakar, and R. Sarkar, "A feature selection-aided deep learning based deepfake video detection method," *Multimedia Tools and Applications*, pp. 1–24, 2025.

[50] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey Wolf Optimizer," *Advances in engineering software*, vol. 69, pp. 46–61, 2014.

[51] B. Doğan and T. Ölmez, "A new metaheuristic for numerical function optimization: Vortex Search algorithm," *Information Sciences*, vol. 293, pp. 125–145, 2015.

[52] S. Mohiuddin, S. Malakar, and R. Sarkar, "An ensemble approach to detect copy-move forgery in videos," *Multimedia Tools and Applications*, vol. 82, no. 16, pp. 24269–24288, 2023.

[53] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE transactions on image processing*, vol. 13, no. 4, pp. 600–612, 2004.

[54] H. Chen, Z. Chen, X. Zeng, W. Fan, and Z. Xiong, "A novel reversible semi-fragile watermarking algorithm of mpeg-4 video for content authentication," in *2008 Second international symposium on intelligent information technology application*, vol. 3, pp. 37–41, IEEE, 2008.

[55] J. Y. Park, J. H. Lim, G. S. Kim, and C. S. Won, "Invertible semi-fragile watermarking algorithm distinguishing mpeg-2 compression from malicious manipulation," in *2002 Digest of Technical Papers. International Conference on Consumer Electronics (IEEE Cat. No. 02CH37300)*, pp. 18–19, IEEE, 2002.

[56] N. Antony and B. R. Devassy, "Implementation of image/video copy-move forgery detection using brute-force matching," in *2018 2nd International conference on trends in electronics and informatics (ICOEI)*, pp. 1085–1090, IEEE, 2018.

[57] J. Bakas, R. Naskar, and R. Dixit, "Detection and localization of inter-frame video forgeries based on inconsistency in correlation distribution between haralick coded frames," *Multimedia Tools and Applications*, vol. 78, pp. 4905–4935, 2019.

[58] R. Davarzani, K. Yaghmaie, S. Mozaffari, and M. Tapak, "Copy-move forgery detection using multiresolution local binary patterns," *Forensic science international*, vol. 231, no. 1-3, pp. 61–72, 2013.

[59] M. Begum and M. S. Uddin, "Digital image watermarking techniques: a review," *Information*, vol. 11, no. 2, p. 110, 2020.

[60] L. Sharma, A. Anand, N. K. Trivedi, M. Sharma, and J. Singh, "Digital video watermarking: features, techniques, and challenges," *Annals of the Romanian Society for Cell Biology*, vol. 25, no. 2, pp. 3376–3385, 2021.

[61] T. Anmol and K. Sitara, "Video source camera identification using fusion of texture features and noise fingerprint," *Forensic Science International: Digital Investigation*, vol. 49, p. 301746, 2024.

[62] A. Bruno, P. Capasso, G. Cattaneo, U. F. Petrillo, and R. Improta, "A novel image dataset for source camera identification and image based recognition systems," *Multimedia Tools and Applications*, vol. 82, no. 8, pp. 11221–11237, 2023.

[63] D. Singh and S. K. Singh, "Effective self-embedding watermarking scheme for image tampered detection and localization with recovery capability," *Journal of Visual Communication and Image Representation*, vol. 38, pp. 775–789, 2016.

[64] D. Bansal and M. Mathuria, "Color image dual watermarking using dct and dwt combine approach," in *2017 International Conference on Trends in Electronics and Informatics (ICEI)*, pp. 630–634, IEEE, 2017.

[65] T. Dutta, A. Sur, and S. Nandi, "A robust compressed domain video watermarking in p-frames with controlled bit rate increase," in *2013 national conference on communications (NCC)*, pp. 1–5, IEEE, 2013.

[66] N. Saikia and P. K. Bora, "Video authentication using temporal wavelet transform," in *15th International Conference on Advanced Computing and Communications (ADCOM 2007)*, pp. 648–653, IEEE, 2007.

[67] S. Bayram, H. Sencar, and N. Memon, "Source camera identification based on cfa interpolation," in *IEEE International Conference on Image Processing 2005*, vol. 3, pp. 69–72, IEEE, 2005.

[68] H. Farid, "Digital image forensics," *Scientific American*, vol. 300, no. 6, pp. 66–71, 2009.

[69] K. Kurosawa, K. Kuroki, and N. Saitoh, "Ccd fingerprint method-identification of a video camera from videotaped images," in *Proceedings 1999 international conference on image processing (Cat. 99CH36348)*, vol. 3, pp. 537–540, IEEE, 1999.

[70] J. Lukas, J. Fridrich, and M. Goljan, "Digital camera identification from sensor pattern noise," *IEEE Transactions on Information Forensics and Security*, vol. 1, no. 2, pp. 205–214, 2006.

[71] O. M. Al-Qershi and B. E. Khoo, "Passive detection of copy-move forgery in digital images: State-of-the-art," *Forensic science international*, vol. 231, no. 1-3, pp. 284–295, 2013.

[72] X. Feng and G. Doërr, "Jpeg recompression detection," in *Media forensics and security II*, vol. 7541, pp. 188–199, SPIE, 2010.

[73] T. Pevny and J. Fridrich, "Detection of double-compression in jpeg images for applications in steganography," *IEEE Transactions on information forensics and security*, vol. 3, no. 2, pp. 247–258, 2008.

[74] J. Lukáš and J. Fridrich, "Estimation of primary quantization matrix in double compressed jpeg images," in *Proc. Digital forensic research workshop*, pp. 5–8, 2003.

[75] W. Wang and H. Farid, "Exposing digital forgeries in video by detecting double mpeg compression," in *Proceedings of the 8th workshop on Multimedia and security*, pp. 37–47, 2006.

[76] W. Wang and H. Farid, "Exposing digital forgeries in video by detecting double quantization," in *Proceedings of the 11th ACM workshop on Multimedia and security*, pp. 39–48, 2009.

[77] T. Sun, W. Wang, and X. Jiang, "Exposing video forgeries by detecting mpeg double compression," in *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1389–1392, IEEE, 2012.

[78] W. Chen and Y. Q. Shi, "Detection of double mpeg compression based on first digit statistics," in *Digital Watermarking: 7th International Workshop, IWDW 2008, Busan, Korea, November 10-12, 2008. Selected Papers 7*, pp. 16–30, Springer, 2009.

[79] Y.-F. Hsu and S.-F. Chang, "Camera response functions for image forensics: an automatic algorithm for splicing detection," *IEEE Transactions on Information Forensics and Security*, vol. 5, no. 4, pp. 816–825, 2010.

[80] D. Fu, Y. Q. Shi, and W. Su, "Detection of image splicing based on hilbert-huang transform and moments of characteristic functions with wavelet decomposition," in *Digital Watermarking: 5th International Workshop, IWDW 2006, Jeju Island, Korea, November 8-10, 2006. Proceedings 5*, pp. 177–187, Springer, 2006.

[81] X. Li, T. Jing, and X. Li, "Image splicing detection based on moment features and hilbert-huang transform," in *2010 IEEE international conference on information theory and information security*, pp. 1127–1130, IEEE, 2010.

[82] Y. Q. Shi, C. Chen, and W. Chen, "A natural image model approach to splicing detection," in *Proceedings of the 9th workshop on Multimedia & security*, pp. 51–62, 2007.

[83] J. Dong, W. Wang, T. Tan, and Y. Q. Shi, "Run-length and edge statistics based approach for image splicing detection," in *Digital Watermarking: 7th International Workshop, IWDW 2008, Busan, Korea, November 10-12, 2008. Selected Papers 7*, pp. 76–87, Springer, 2009.

[84] H. Kaur, J. Saxena, and S. Singh, "Key-point based copy-move forgery detection and their hybrid methods: A review," *Journal of The International Association of Advanced Technology and Science*, vol. 16, no. 02, 2015.

[85] S. Teerakanok and T. Uehara, "Copy-move forgery detection using glcm-based rotation-invariant feature: A preliminary research," in *2018 IEEE 42nd annual computer software and applications conference (COMPSAC)*, vol. 2, pp. 365–369, IEEE, 2018.

[86] M. Emam, Q. Han, L. Yu, and H. Zhang, "A keypoint-based region duplication forgery detection algorithm," *IEICE TRANSACTIONS on Information and Systems*, vol. 99, no. 9, pp. 2413–2416, 2016.

[87] I. Amerini, L. Ballan, R. Caldelli, A. Del Bimbo, and G. Serra, "Geometric tampering estimation by means of a sift-based forensic analysis," in *2010 IEEE international conference on acoustics, speech and signal processing*, pp. 1702–1705, IEEE, 2010.

[88] N. B. A. Warif, A. W. A. Wahab, M. Y. I. Idris, R. Salleh, and F. Othman, "Sift-symmetry: a robust detection method for copy-move forgery with reflection attack," *Journal of Visual Communication and Image Representation*, vol. 46, pp. 219–232, 2017.

[89] S. T. Babu and C. S. Rao, "An optimized technique for copy–move forgery localization using statistical features," *ICT Express*, vol. 8, no. 2, pp. 244–249, 2022.

[90] K. M. Hosny, A. M. Mortda, M. M. Fouda, and N. A. Lashin, "An efficient cnn model to detect copy-move image forgery," *IEEE Access*, vol. 10, pp. 48622–48632, 2022.

[91] R. Zhang and J. Ni, "A dense u-net with cross-layer intersection for detection and localization of image forgery," in *ICASSP 2020-2020 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pp. 2982–2986, IEEE, 2020.

[92] Y. Liu, C. Xia, X. Zhu, and S. Xu, "Two-stage copy-move forgery detection with self deep matching and proposal superglue," *IEEE Transactions on Image Processing*, vol. 31, pp. 541–555, 2021.

[93] J.-L. Zhong and C.-M. Pun, "An end-to-end dense-inceptionnet for image copy-move forgery detection," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 2134–2146, 2019.

[94] M. Stamm and K. R. Liu, "Blind forensics of contrast enhancement in digital images," in *2008 15th IEEE International Conference on Image Processing*, pp. 3112–3115, IEEE, 2008.

[95] G. Cao, Y. Zhao, and R. Ni, "Detection of image sharpening based on histogram aberration and ringing artifacts," in *2009 IEEE International Conference on Multimedia and Expo*, pp. 1026–1029, IEEE, 2009.

[96] F. Ding, G. Zhu, J. Yang, J. Xie, and Y.-Q. Shi, "Edge perpendicular binary coding for usm sharpening detection," *IEEE signal processing letters*, vol. 22, no. 3, pp. 327–331, 2014.

[97] Z. Zhang, J. Hou, Q. Ma, and Z. Li, "Efficient video frame insertion and deletion detection based on inconsistency of correlations between local binary pattern coded frames," *Security and Communication networks*, vol. 8, no. 2, pp. 311–320, 2015.

[98] W. Wang and H. Farid, "Exposing digital forgeries in video by detecting duplication," in *Proceedings of the 9th workshop on Multimedia & security*, pp. 35–42, 2007.

[99] Q. Li, R. Wang, and D. Xu, "An inter-frame forgery detection algorithm for surveillance video," *Information*, vol. 9, no. 12, p. 301, 2018.

[100] X. Jin, Y. Su, and P. Jing, "Video frame deletion detection based on time–frequency analysis," *Journal of Visual Communication and Image Representation*, vol. 83, p. 103436, 2022.

[101] J. Bakas, R. Naskar, and S. Bakshi, "Detection and localization of inter-frame forgeries in videos based on macroblock variation and motion vector analysis," *Computers & Electrical Engineering*, vol. 89, p. 106929, 2021.

[102] S. Kingra, N. Aggarwal, and R. D. Singh, "Inter-frame forgery detection in h. 264 videos using motion and brightness gradients," *Multimedia Tools and Applications*, vol. 76, pp. 25767–25786, 2017.

[103] D.-N. Zhao, R.-K. Wang, and Z.-M. Lu, "Inter-frame passive-blind forgery detection for video shot based on similarity analysis," *Multimedia Tools and Applications*, vol. 77, pp. 25389–25408, 2018.

[104] N. A. Shelke and S. S. Kasana, "Multiple forgeries identification in digital video based on correlation consistency between entropy coded frames," *Multimedia Systems*, vol. 28, no. 1, pp. 267–280, 2022.

[105] N. Bansal, T. Aljrees, D. P. Yadav, K. U. Singh, A. Kumar, G. K. Verma, and T. Singh, "Real-time advanced computational intelligence for deep fake video detection," *Applied Sciences*, vol. 13, no. 5, p. 3095, 2023.

[106] X. Yang, Y. Li, and S. Lyu, "Exposing deep fakes using inconsistent head poses," in *ICASSP 2019-2019 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pp. 8261–8265, IEEE, 2019.

[107] Z. Guo, G. Yang, J. Chen, and X. Sun, "Fake face detection via adaptive manipulation traces extraction network," *Computer Vision and Image Understanding*, vol. 204, p. 103170, 2021.

[108] W. Lu, L. Liu, B. Zhang, J. Luo, X. Zhao, Y. Zhou, and J. Huang, "Detection of deepfake videos using long-distance attention," *IEEE Transactions on Neural Networks and Learning Systems*, 2023.

[109] R. Xia, D. Liu, J. Li, L. Yuan, N. Wang, and X. Gao, "Mmnet: Multi-collaboration and multi-supervision network for sequential deepfake detection," *IEEE Transactions on Information Forensics and Security*, 2024.

[110] H. Wang, Z. Liu, and S. Wang, "Exploiting complementary dynamic incoherence for deepfake video detection," *IEEE Transactions on Circuits and Systems for Video Technology*, 2023.

[111] E. Amiri, A. Mosallanejad, and A. Sheikhahmadi, "Cfdmi-sec: An optimal model for copy-move forgery detection of medical image using sift, eom and chm," *Plos one*, vol. 19, no. 7, p. e0303332, 2024.

[112] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, pp. 91–110, 2004.

[113] A. Diwan, R. Mahadeva, and V. Gupta, "Advancing copy-move manipulation detection in complex image scenarios through multiscale detector," *IEEE Access*, 2024.

[114] H. Chen, X. Yang, and Y. Lyu, "Copy-move forgery detection based on keypoint clustering and similar neighborhood search algorithm," *IEEE Access*, vol. 8, pp. 36863–36875, 2020.

[115] C.-C. Chen, W.-Y. Lu, and C.-H. Chou, "Rotational copy-move forgery detection using sift and region growing strategies," *Multimedia Tools and Applications*, vol. 78, pp. 18293–18308, 2019.

[116] X.-y. Wang, L.-x. Jiao, X.-b. Wang, H.-y. Yang, and P.-p. Niu, "Copy-move forgery detection based on compact color content descriptor and delaunay triangle matching," *Multimedia Tools and Applications*, vol. 78, no. 2, pp. 2311–2344, 2019.

[117] P.-p. Niu, C. Wang, W. Chen, H. Yang, and X. Wang, "Fast and effective keypoint-based image copy-move forgery detection using complex-valued moment invariants," *Journal of Visual Communication and Image Representation*, vol. 77, p. 103068, 2021.

[118] G. Muzaffer and G. Ulutas, "A fast and effective digital image copy move forgery detection with binarized sift," in *2017 40th International Conference on Telecommunications and Signal Processing (TSP)*, pp. 595–598, IEEE, 2017.

[119] L. Xie, Q. Tian, J. Wang, and B. Zhang, "Image classification with max-sift descriptors," in *International conference on acoustics, speech and signal processing*, 2015.

[120] F. Yang, J. Li, W. Lu, and J. Weng, "Copy-move forgery detection based on hybrid features," *Engineering Applications of Artificial Intelligence*, vol. 59, pp. 73–83, 2017.

[121] X.-Y. Wang, S. Li, Y.-N. Liu, Y. Niu, H.-Y. Yang, and Z.-l. Zhou, "A new keypoint-based copy-move forgery detection for small smooth regions," *Multimedia Tools and Applications*, vol. 76, pp. 23353–23382, 2017.

[122] X.-y. Wang, X.-q. Wang, P.-p. Niu, and H.-y. Yang, "Accurate and robust image copy-move forgery detection using adaptive keypoints and fqgpcet-glcm feature," *Multimedia Tools and Applications*, vol. 83, no. 1, pp. 2203–2235, 2024.

[123] C. Lin, W. Lu, X. Huang, K. Liu, W. Sun, H. Lin, and Z. Tan, "Copy-move forgery detection using combined features and transitive matching," *Multimedia Tools and Applications*, vol. 78, pp. 30081–30096, 2019.

[124] X. Pan and S. Lyu, "Region duplication detection using image feature matching," *IEEE Transactions on Information Forensics and Security*, vol. 5, no. 4, pp. 857–867, 2010.

[125] I. Amerini, L. Ballan, R. Caldelli, A. Del Bimbo, L. Del Tongo, and G. Serra, "Copy-move forgery detection and localization by means of robust clustering with j-linkage," *Signal Processing: Image Communication*, vol. 28, no. 6, pp. 659–669, 2013.

[126] G. Jin and X. Wan, "An improved method for sift-based copy–move forgery detection using non-maximum value suppression and optimized j-linkage," *Signal Processing: Image Communication*, vol. 57, pp. 113–125, 2017.

[127] V. Anand, M. F. Hashmi, and A. G. Keskar, "A copy move forgery detection to overcome sustained attacks using dyadic wavelet transform and sift methods," in *Intelligent Information and Database Systems: 6th Asian Conference, ACIIDS 2014, Bangkok, Thailand, April 7-9, 2014, Proceedings, Part I 6*, pp. 530–542, Springer, 2014.

[128] J. Gong and J. Guo, "Exposing region duplication through local geometrical color invariant features," *Journal of Electronic Imaging*, vol. 24, no. 3, pp. 033010–033010, 2015.

[129] P. P. Panzade, C. S. Prakash, and S. Maheshkar, "Copy-move forgery detection by using hsv preprocessing and keypoint extraction," in *2016 Fourth International Conference on Parallel, Distributed and Grid Computing (PDGC)*, pp. 264–269, IEEE, 2016.

[130] F. Zhao, W. Shi, B. Qin, and B. Liang, "Analysis of sift method based on swarm intelligent algorithms for copy-move forgery detection," in *Security, Privacy, and Anonymity in Computation, Communication, and Storage: 9th International Conference, SpaCCS 2016, Zhangjiajie, China, November 16-18, 2016, Proceedings 9*, pp. 478–490, Springer, 2016.

[131] S. Wenchang, Z. Fei, Q. Bo, and L. Bin, "Improving image copy-move forgery detection with particle swarm optimization techniques," *China Communications*, vol. 13, no. 1, pp. 139–149, 2016.

[132] G. Rajput, S. D. Dabhole, *et al.*, "Modified keypoint-based copy move area detection," *Procedia Computer Science*, vol. 235, pp. 3389–3396, 2024.

[133] P. Deb, N. Kar, K. L. Hassan, and B. Biswas, "Advanced copy-move forgery detection: utilizing akaze in conjunction with sift algorithm for image forensics," *Microsystem Technologies*, pp. 1–9, 2024.

[134] B. Shivakumar and S. S. Baboo, "Detection of region duplication forgery in digital images using surf," *International Journal of Computer Science Issues (IJCSI)*, vol. 8, no. 4, p. 199, 2011.

[135] B. B. MP and A. Kumar, "Copy-move forgery detection using segmentation," in *2017 11th International Conference on Intelligent Systems and Control (ISCO)*, pp. 224–228, IEEE, 2017.

[136] P. Mishra, N. Mishra, S. Sharma, and R. Patel, "Region duplication forgery detection technique based on surf and hac," *The Scientific World Journal*, vol. 2013, no. 1, p. 267691, 2013.

[137] E. Silva, T. Carvalho, A. Ferreira, and A. Rocha, "Going deeper into copy-move forgery detection: Exploring image telltales via multi-scale analysis and voting processes," *Journal of Visual Communication and Image Representation*, vol. 29, pp. 16–32, 2015.

[138] B. Yang, X. Sun, X. Xin, W. Hu, and Y. Wu, "Image copy–move forgery detection based on sped-up robust features descriptor and adaptive minimal–maximal suppression," *Journal of Electronic Imaging*, vol. 24, no. 6, pp. 063016–063016, 2015.

[139] K. Sunitha, A. Krishna, and B. Prasad, "Copy-move tampering detection using keypoint based hybrid feature extraction and improved transformation model," *Applied Intelligence*, vol. 52, no. 13, pp. 15405–15416, 2022.

[140] M. M. A. Alhaidery, A. H. Taherinia, and H. I. Shahadi, "A robust detection and localization technique for copy-move forgery in digital images," *Journal of King Saud University-Computer and Information Sciences*, vol. 35, no. 1, pp. 449–461, 2023.

[141] L. Chen, W. Lu, J. Ni, W. Sun, and J. Huang, "Region duplication detection based on harris corner points and step sector statistics," *Journal of Visual Communication and Image Representation*, vol. 24, no. 3, pp. 244–254, 2013.

[142] J. Zhao and W. Zhao, "Passive forensics for region duplication image forgery based on harris feature points and local binary patterns," *Mathematical Problems in Engineering*, vol. 2013, no. 1, p. 619564, 2013.

[143] X. Wang, G. He, C. Tang, Y. Han, and S. Wang, "Keypoints-based image passive forensics method for copy-move attacks," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 30, no. 03, p. 1655008, 2016.

[144] D. M. Uliyan, H. A. Jalab, A. W. Abdul Wahab, and S. Sadeghi, "Image region duplication forgery detection based on angular radial partitioning and harris key-points," *Symmetry*, vol. 8, no. 7, p. 62, 2016.

[145] J. Fridrich, D. Soukal, J. Lukas, *et al.*, "Detection of copy-move forgery in digital images," in *Proceedings of digital forensic research workshop*, vol. 3, pp. 652–63, Cleveland, OH, 2003.

[146] A. C. Popescu and H. Farid, "Exposing digital forgeries by detecting duplicated image regions," *Computer Science Technical Report*, 2004.

[147] S. T. Babu and C. S. Rao, "Efficient detection of copy-move forgery using polar complex exponential transform and gradient direction pattern," *Multimedia Tools and Applications*, vol. 82, no. 7, pp. 10061–10075, 2023.

[148] S. T. Babu and C. S. Rao, "Copy-move forgery verification in images using local feature extractors and optimized classifiers," *Big Data Mining and Analytics*, vol. 6, no. 3, pp. 347–360, 2023.

[149] B. Chen, M. Yu, Q. Su, and L. Li, "Fractional quaternion cosine transform and its application in color image copy-move forgery detection," *Multimedia Tools and Applications*, vol. 78, no. 7, pp. 8057–8073, 2019.

[150] B. Chen, M. Yu, Q. Su, H. J. Shim, and Y.-Q. Shi, "Fractional quaternion zernike moments for robust color image copy-move forgery detection," *IEEE Access*, vol. 6, pp. 56637–56646, 2018.

[151] T. Mahmood, Z. Mehmood, M. Shah, and T. Saba, "A robust technique for copy-move forgery detection and localization in digital images via stationary wavelet and discrete cosine transform," *Journal of Visual Communication and Image Representation*, vol. 53, pp. 202–214, 2018.

[152] S. Dua, J. Singh, and H. Parthasarathy, "Detection and localization of forgery using statistics of dct and fourier components," *Signal processing: image communication*, vol. 82, p. 115778, 2020.

[153] N. K. Rathore, N. K. Jain, P. K. Shukla, U. Rawat, and R. Dubey, "Image forgery detection using singular value decomposition with some attacks," *National Academy Science Letters*, vol. 44, no. 4, pp. 331–338, 2021.

[154] S. Kumar, S. Mukherjee, and A. K. Pal, "An improved reduced feature-based copy-move forgery detection technique," *Multimedia Tools and Applications*, vol. 82, no. 1, pp. 1431–1456, 2023.

[155] S. Ganguly, S. Mandal, S. Malakar, and R. Sarkar, "Copy-move forgery detection using local tetra pattern based texture descriptor," *Multimedia Tools and Applications*, vol. 82, no. 13, pp. 19621–19642, 2023.

[156] Y. Rao and J. Ni, "A deep learning approach to detection of splicing and copy-move forgeries in images," in *2016 IEEE international workshop on information forensics and security (WIFS)*, pp. 1–6, IEEE, 2016.

[157] S. Nikalje and M. V. Mane, "Copy-move and image splicing forgery detection based on convolution neural network," in *2022 Third International Conference on Intelligent Computing Instrumentation and Control Technologies (ICICICT)*, pp. 391–395, IEEE, 2022.

[158] M. A. Elaskily, H. A. Elnemr, A. Sedik, M. M. Dessouky, G. M. El Banby, O. A. Elshakankiry, A. A. Khalaf, H. K. Aslan, O. S. Faragallah, and F. E. Abd El-Samie, "A novel deep learning framework for copy-moveforgery detection in images," *Multimedia Tools and Applications*, vol. 79, pp. 19167–19192, 2020.

[159] Y. Zhu, C. Chen, G. Yan, Y. Guo, and Y. Dong, "Ar-net: Adaptive attention and residual refinement network for copy-move forgery detection," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 10, pp. 6714–6723, 2020.

[160] M. Sabeena and L. Abraham, "Convolutional block attention based network for copy-move image forgery detection," *Multimedia Tools and Applications*, vol. 83, no. 1, pp. 2383–2405, 2024.

[161] A. Islam, C. Long, A. Basharat, and A. Hoogs, "Doa-gan: Dual-order attentive generative adversarial network for image copy-move forgery detection and localization," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 4676–4685, 2020.

[162] S. Weng, T. Zhu, T. Zhang, and C. Zhang, "Ucm-net: A u-net-like tampered-region-related framework for copy-move forgery detection," *IEEE Transactions on Multimedia*, 2023.

[163] Y. Wu, W. Abd-Almageed, and P. Natarajan, "Busternet: Detecting copy-move image forgery with source/target localization," in *Proceedings of the European conference on computer vision (ECCV)*, pp. 168–184, 2018.

[164] B. Chen, W. Tan, G. Coatrieux, Y. Zheng, and Y.-Q. Shi, "A serial image copy-move forgery localization scheme with source/target distinguishment," *IEEE Transactions on Multimedia*, vol. 23, pp. 3506–3517, 2020.

[165] K. Zhao, X. Yuan, T. Liu, Y. Xiang, Z. Xie, G. Huang, and L. Feng, "Camu-net: Copy-move forgery detection utilizing coordinate attention and multi-scale feature fusion-based up-sampling," *Expert Systems with Applications*, vol. 238, p. 121918, 2024.

[166] F. F. Niloy, K. K. Bhaumik, and S. S. Woo, "Cfl-net: Image forgery localization using contrastive learning," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 4642–4651, 2023.

[167] P. Kumar, A. Singhal, S. Mehta, and A. Mittal, "Real-time moving object detection algorithm on high-resolution videos using gpus," *Journal of Real-Time Image Processing*, vol. 11, pp. 93–109, 2016.

[168] A. Anjum, T. Abdullah, M. F. Tariq, Y. Baltaci, and N. Antonopoulos, "Video stream analysis in clouds: An object detection and classification framework for high performance video analytics," *IEEE Transactions on Cloud Computing*, vol. 7, no. 4, pp. 1152–1167, 2016.

[169] Y. Choi, M. Choi, M. Kim, J.-W. Ha, S. Kim, and J. Choo, "Stargan: Unified generative adversarial networks for multi-domain image-to-image translation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 8789–8797, 2018.

[170] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 4401–4410, 2019.

[171] R. Durall, M. Keuper, F.-J. Pfreundt, and J. Keuper, "Unmasking deepfakes with simple features," *arXiv preprint arXiv:1911.00686*, 2019.

[172] M. Koopman, A. M. Rodriguez, and Z. Geradts, "Detection of deepfake video manipulation," in *The 20th Irish machine vision and image processing conference (IMVIP)*, pp. 133–136, 2018.

[173] B. Bayar and M. C. Stamm, "Constrained convolutional neural networks: A new approach towards general purpose image manipulation detection," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 11, pp. 2691–2706, 2018.

[174] A. Rossler, D. Cozzolino, L. Verdoliva, C. Riess, J. Thies, and M. Nießner, "Faceforensics++: Learning to detect manipulated facial images," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1–11, 2019.

[175] Y. Li, X. Yang, P. Sun, H. Qi, and S. Lyu, "Celeb-df: A large-scale challenging dataset for deepfake forensics," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3207–3216, 2020.

[176] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1–9, 2015.

[177] D. Nguyen, N. Mejri, I. P. Singh, P. Kuleshova, M. Astrid, A. Kacem, E. Ghorbel, and D. Aouada, "Laa-net: Localized artifact attention network for high-quality deepfakes detection," *arXiv preprint arXiv:2401.13856*, 2024.

[178] T. Wang, H. Cheng, K. P. Chow, and L. Nie, "Deep convolutional pooling transformer for deepfake detection," *ACM Transactions on Multimedia Computing, Communications and Applications*, vol. 19, no. 6, pp. 1–20, 2023.

[179] Y.-J. Heo, W.-H. Yeo, and B.-G. Kim, "Deepfake detection algorithm based on improved vision transformer," *Applied Intelligence*, vol. 53, no. 7, pp. 7512–7527, 2023.

[180] S. Wang, D. Zhu, J. Chen, J. Bi, and W. Wang, "Deepfake face discrimination based on self-attention mechanism," *Pattern Recognition Letters*, vol. 183, pp. 92–97, 2024.

[181] P. M. G. I. Reis and R. O. Ribeiro, "A forensic evaluation method for deepfake detection using dcnn-based facial similarity scores," *Forensic Science International*, vol. 358, p. 111747, 2024.

[182] C. Tian, Z. Luo, G. Shi, and S. Li, "Frequency-aware attentional feature fusion for deepfake detection," in *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1–5, IEEE, 2023.

[183] J. Liu, K. Zhu, W. Lu, X. Luo, and X. Zhao, "A lightweight 3d convolutional neural network for deepfake detection," *International Journal of Intelligent Systems*, vol. 36, no. 9, pp. 4990–5004, 2021.

[184] D. Zhang, C. Li, F. Lin, D. Zeng, and S. Ge, "Detecting deepfake videos with temporal dropout 3dcnn.," in *IJCAI*, pp. 1288–1294, 2021.

[185] K. Lin, W. Han, S. Li, Z. Gu, H. Zhao, and Y. Mei, "Detecting deepfake videos using spatiotemporal trident network," *ACM Transactions on Multimedia Computing, Communications and Applications*, vol. 20, no. 11, pp. 1–20, 2024.

[186] A. Almestekawy, H. H. Zayed, and A. Taha, "Deepfake detection: Enhancing performance with spatiotemporal texture and deep learning feature fusion," *Egyptian Informatics Journal*, vol. 27, p. 100535, 2024.

[187] R. D. Singh and N. Aggarwal, "Video content authentication techniques: a comprehensive survey," *Multimedia Systems*, vol. 24, no. 2, pp. 211–240, 2018.

[188] O. I. Al-Sanjary and G. Sulong, "Detection of video forgery: A review of literature," *Journal of Theoretical & Applied Information Technology*, vol. 74, no. 2, pp. 208–220, 2015.

[189] J. Kharat and S. Chougule, "A passive blind forgery detection technique to identify frame duplication attack," *Multimedia Tools and Applications*, vol. 79, no. 11, pp. 8107–8123, 2020.

[190] S. Jia, Z. Xu, H. Wang, C. Feng, and T. Wang, "Coarse-to-fine copy-move forgery detection for video forensics," *IEEE Access*, vol. 6, pp. 25323–25335, 2018.

[191] Q. Wang, Z. Li, Z. Zhang, and Q. Ma, "Video inter-frame forgery identification based on optical flow consistency," *Sensors & Transducers*, vol. 166, no. 3, p. 229, 2014.

[192] R. D. Singh and N. Aggarwal, "Optical flow and prediction residual based hybrid forensic system for inter-frame tampering detection," *Journal of Circuits, Systems and Computers*, vol. 26, no. 07, p. 1750107, 2017.

[193] L. Yu, H. Wang, Q. Han, X. Niu, S.-M. Yiu, J. Fang, and Z. Wang, "Exposing frame deletion by detecting abrupt changes in video streams," *Neurocomputing*, vol. 205, pp. 84–91, 2016.

[194] M. C. Stamm, W. S. Lin, and K. R. Liu, "Temporal forensics and anti-forensics for motion compensated video," *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 4, pp. 1315–1329, 2012.

[195] S. M. Fadl, Q. Han, and Q. Li, "Authentication of surveillance videos: detecting frame duplication based on residual frame," *Journal of forensic sciences*, vol. 63, no. 4, pp. 1099–1109, 2018.

[196] G. Ulutas, B. Ustubioglu, M. Ulutas, and V. V. Nabiyev, "Frame duplication detection based on bow model," *Multimedia Systems*, vol. 24, pp. 549–567, 2018.

[197] S. Chen, S. Tan, B. Li, and J. Huang, "Automatic detection of object-based forgery in advanced video," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 26, no. 11, pp. 2138–2151, 2015.

[198] M. A. Fayyaz, A. Anjum, S. Ziauddin, A. Khan, and A. Sarfaraz, "An improved surveillance video forgery detection technique using sensor pattern noise and correlation of noise residues," *Multimedia Tools and Applications*, vol. 79, pp. 5767–5788, 2020.

[199] Z. Li, Z. Zhang, S. Guo, and J. Wang, "Video inter-frame forgery identification based on the consistency of quotient of mssim," *Security and Communication Networks*, vol. 9, no. 17, pp. 4548–4556, 2016.

[200] G. Singh and K. Singh, "Video frame and region duplication forgery detection based on correlation coefficient and coefficient of variation," *Multimedia Tools and Applications*, vol. 78, no. 9, pp. 11527–11562, 2019.

[201] S. M. Fadl, Q. Han, and Q. Li, "Inter-frame forgery detection based on differential energy of residue," *IET Image Processing*, vol. 13, no. 3, pp. 522–528, 2019.

[202] W. Wang, Z. Zhao, N. Sebe, and B. Lepri, "Turn fake into real: Adversarial head turn attacks against deepfake detection," *arXiv preprint arXiv:2309.01104*, 2023.

[203] A. Diwan, D. Kumar, R. Mahadeva, H. Perera, and J. Alawatugoda, "Unveiling copy-move forgeries: enhancing detection with superpoint keypoint architecture," *IEEE Access*, vol. 11, pp. 86132–86148, 2023.

[204] J. Wang, X. Gao, J. Nie, X. Wang, L. Huang, W. Nie, M. Jiang, and Z. Wei, "Strong robust copy-move forgery detection network based on layer-by-layer de-

coupling refinement," *Information Processing & Management*, vol. 61, no. 3, p. 103685, 2024.

[205] Y. Shi, S. Weng, L. Yu, and L. Li, "Lightweight and high-precision network for image copy-move forgery detection," *IEEE Signal Processing Letters*, 2024.

[206] Y. Shi, S. Weng, L. Yu, and L. Li, "A copy-move forgery detection network based on selective sampling attention and low-cost two-step self-correlation calculation," *IEEE Transactions on Multimedia*, 2025.

[207] E. Liang, K. Zhang, Z. Hua, Y. Li, and X. Jia, "Transcmfd: An adaptive transformer for copy-move forgery detection," *Neurocomputing*, p. 130110, 2025.

[208] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

[209] D. Tralic, I. Zupancic, S. Grgic, and M. Grgic, "Comofod—new database for copy-move forgery detection," in *Proceedings ELMAR-2013*, pp. 49–54, IEEE, 2013.

[210] B. Wen, Y. Zhu, R. Subramanian, T.-T. Ng, X. Shen, and S. Winkler, "Coverage—a novel database for copy-move forgery detection," in *2016 IEEE international conference on image processing (ICIP)*, pp. 161–165, IEEE, 2016.

[211] J. Dong, W. Wang, and T. Tan, "Casia image tampering detection evaluation database," in *2013 IEEE China summit and international conference on signal and information processing*, pp. 422–426, IEEE, 2013.

[212] M. Aria, M. Hashemzadeh, and N. Farajzadeh, "Qdl-cmfd: a quality-independent and deep learning-based copy-move image forgery detection method," *Neurocomputing*, vol. 511, pp. 213–236, 2022.

[213] X. Lin, S. Wang, J. Deng, Y. Fu, X. Bai, X. Chen, X. Qu, and W. Tang, "Image manipulation detection by multiple tampering traces and edge artifact enhancement," *Pattern Recognition*, vol. 133, p. 109026, 2023.

[214] Y. Li and J. Zhou, "Fast and effective image copy-move forgery detection via hierarchical feature point matching," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 5, pp. 1307–1322, 2018.

[215] Y. Huang, S. Bian, H. Li, C. Wang, and K. Li, "Ds-unet: A dual streams unet for refined image forgery localization," *Information Sciences*, vol. 610, pp. 73–89, 2022.

[216] X. Liu, Y. Liu, J. Chen, and X. Liu, "Pscc-net: Progressive spatio-channel correlation network for image manipulation detection and localization," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 32, no. 11, pp. 7505–7517, 2022.

[217] S. Tinnathi and G. Sudhavani, "An efficient copy move forgery detection using adaptive watershed segmentation with agso and hybrid feature extraction," *Journal of Visual Communication and Image Representation*, vol. 74, p. 102966, 2021.

[218] Y. Li, M.-C. Chang, and S. Lyu, "In ictu oculi: Exposing ai created fake videos by detecting eye blinking," in *2018 IEEE International Workshop on Information Forensics and Security (WIFS)*, pp. 1–7, IEEE, 2018.

[219] T. Jung, S. Kim, and K. Kim, "Deepvision: deepfakes detection using human eye blinking pattern," *IEEE Access*, vol. 8, pp. 83144–83154, 2020.

[220] R. Tolosana, S. Romero-Tapiador, J. Fierrez, and R. Vera-Rodriguez, "Deepfakes evolution: Analysis of facial regions and fake detection performance," in *International Conference on Pattern Recognition*, pp. 442–456, Springer, 2021.

[221] F. Matern, C. Riess, and M. Stamminger, "Exploiting visual artifacts to expose deepfakes and face manipulations," in *2019 IEEE Winter Applications of Computer Vision Workshops (WACVW)*, pp. 83–92, IEEE, 2019.

[222] S. Walia, K. Kumar, M. Kumar, and X.-Z. Gao, "Fusion of handcrafted and deep features for forgery detection in digital images," *IEEE Access*, vol. 9, pp. 99742–99755, 2021.

[223] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," in *International Conference on Learning Representations (ICLR-2021)*, pp. 1–22, 2021.

[224] B. Dolhansky, J. Bitton, B. Pflaum, J. Lu, R. Howes, M. Wang, and C. C. Ferrer, "The DeepFake Detection Challenge (DFDC) Dataset," *arXiv preprint arXiv:2006.07397*, 2020.

[225] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Proceedings of the 2001 IEEE computer society conference on computer vision and pattern recognition. CVPR 2001*, vol. 1, pp. I–I, Ieee, 2001.

[226] B. Dolhansky, R. Howes, B. Pflaum, N. Baram, and C. C. Ferrer, "The deepfake detection challenge (dfdc) preview dataset," *arXiv preprint arXiv:1910.08854*, 2019.

[227] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao, "Joint face detection and alignment using multitask cascaded convolutional networks," *IEEE Signal Processing Letters*, vol. 23, no. 10, pp. 1499–1503, 2016.

[228] Y. Qian, G. Yin, L. Sheng, Z. Chen, and J. Shao, "Thinking in frequency: Face forgery detection by mining frequency-aware clues," in *European Conference on Computer Vision*, pp. 86–103, Springer, 2020.

[229] D. A. Coccomini, N. Messina, C. Gennaro, and F. Falchi, "Combining EfficientNet and Vision Transformers for Video Deepfake Detection," in *International Conference on Image Analysis and Processing*, pp. 219–229, Springer, 2022.

[230] Y.-J. Heo, Y.-J. Choi, Y.-W. Lee, and B.-G. Kim, "Deepfake Detection Scheme Based on Vision Transformer and Distillation," *arXiv preprint arXiv:2104.01353*, 2021.

[231] S. Majumder, S. Ghosh, S. Malakar, R. Sarkar, and M. Nasipuri, "A voting-based technique for word spotting in handwritten document images," *Multimedia Tools and Applications*, vol. 80, no. 8, pp. 12411–12434, 2021.

[232] E. Tola, V. Lepetit, and P. Fua, "DAISY: An Efficient Dense Descriptor Applied to Wide-Baseline Stereo," *IEEE transactions on pattern analysis and machine intelligence*, vol. 32, no. 5, pp. 815–830, 2009.

[233] H. Alzer, "ON SOME INEQUALITIES FOR THE INCOMPLETE GAMMA FUNCTION," *Mathematics of Computation*, vol. 66, no. 218, pp. 771–778, 1997.

[234] Z. Wang, G. Wu, and Z. Wan, "A novel hybrid vortex search and artificial bee colony algorithm for numerical optimization problems," *Wuhan University Journal of Natural Sciences*, vol. 22, no. 4, pp. 295–306, 2017.

[235] S. Malakar, M. Ghosh, S. Bhowmik, R. Sarkar, and M. Nasipuri, "A GA based hierarchical feature selection approach for handwritten word recognition," *Neural Computing and Applications*, vol. 32, pp. 2533–2552, Jan. 2019.

[236] S. Ahmed, K. K. Ghosh, L. Garcia-Hernandez, A. Abraham, and R. Sarkar, "Improved coral reefs optimization with adaptive β-hill climbing for feature selection," *Neural Computing and Applications*, vol. 33, pp. 6467–6486, Oct. 2020.

[237] H. Zhao, W. Zhou, D. Chen, T. Wei, W. Zhang, and N. Yu, "Multi-Attentional Deepfake Detection," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 2185–2194, 2021.

[238] C. Miao, Q. Chu, W. Li, S. Li, Z. Tan, W. Zhuang, and N. Yu, "Learning Forgery Region-Aware and ID-Independent Features for Face Manipulation Detection," *IEEE Transactions on Biometrics, Behavior, and Identity Science*, vol. 4, no. 1, pp. 71–84, 2021.

[239] P. He, X. Jiang, T. Sun, S. Wang, B. Li, and Y. Dong, "Frame-wise detection of relocated i-frames in double compressed h. 264 videos based on convolutional neural network," *Journal of Visual Communication and Image Representation*, vol. 48, pp. 149–158, 2017.

[240] L. Zheng, T. Sun, and Y.-Q. Shi, "Inter-frame video forgery detection based on block-wise brightness variance descriptor," in *International Workshop on Digital Watermarking*, pp. 18–30, Springer, 2014.

[241] J.-P. Jodoin, G.-A. Bilodeau, and N. Saunier, "Urban tracker: Multiple object tracking in urban mixed traffic," in *IEEE Winter Conference on Applications of Computer Vision*, pp. 885–892, IEEE, 2014.

[242] S. Fadl, Q. Han, and Q. Li, "Cnn spatiotemporal features and fusion for surveillance video forgery detection," *Signal Processing: Image Communication*, vol. 90, p. 116066, 2021.

[243] V. Kumar and M. Gaur, "Multiple forgery detection in video using inter-frame correlation distance with dual-threshold," *Multimedia Tools and Applications*, vol. 81, no. 30, pp. 43979–43998, 2022.

[244] H. D. Panchal and H. B. Shah, "Multiple forgery detection in digital video based on inconsistency in video quality assessment attributes," *Multimedia Systems*, vol. 29, no. 4, pp. 2439–2454, 2023.

[245] R. Gowda and D. Pawar, "Deep learning-based forgery identification and localization in videos," *Signal, Image and Video Processing*, vol. 17, no. 5, pp. 2185–2192, 2023.

[246] J. Patel and R. Sheth, "An optimized convolution neural network based inter-frame forgery detection model-a multi-feature extraction framework," *ICTACT J Image Video Process*, vol. 12, pp. 2570–2581, 2021.

[247] S. Li and H. Huo, "Frame deletion detection based on optical flow orientation variation," *IEEE Access*, vol. 9, pp. 37196–37209, 2021.

[248] J. Chao, X. Jiang, and T. Sun, "A novel video inter-frame forgery model detection scheme based on optical flow consistency," in *International workshop on digital watermarking*, pp. 267–281, Springer, 2012.

[249] X. H. Nguyen, Y. Hu, M. A. Amin, G. H. Khan, D.-T. Truong, *et al.*, "Detecting video inter-frame forgeries based on convolutional neural network model," *International Journal of Image, Graphics and Signal Processing*, vol. 10, no. 3, p. 1, 2020.

[250] C. Feng, D. Wu, T. Wu, and L. Wei, "An msdcnn-lstm framework for video frame deletion forensics," *Multimedia Tools and Applications*, vol. 83, no. 29, pp. 72745–72764, 2024.

[251] S. S. Gill, M. Golec, J. Hu, M. Xu, J. Du, H. Wu, G. K. Walia, S. S. Murugesan, B. Ali, M. Kumar, *et al.*, "Edge ai: A taxonomy, systematic review and future directions," *Cluster Computing*, vol. 28, no. 1, pp. 1–53, 2025.

[252] M. M. H. Shuvo and et al., "Efficient acceleration of deep learning inference on resource-constrained edge devices: A review," *Proceedings of the IEEE*, vol. 111, pp. 41–58, 2023.

[253] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding," *International Conference on Learning Representations (ICLR)*, 2016.

[254] A. Van Wynsberghe, "Sustainable ai: Ai for sustainability and the sustainability of ai," *AI and Ethics*, vol. 1, no. 3, pp. 213–218, 2021.

[255] R. Nishant, M. Kennedy, and J. Corbett, "Artificial intelligence for sustainability: Challenges, opportunities, and a research agenda," *International journal of information management*, vol. 53, p. 102104, 2020.

[256] S. Malakar, N. Banerjee, and D. K. Prasad, "Compact representation for memory-efficient storage of images using genetic algorithm-guided key pixel selection," *Engineering Applications of Artificial Intelligence*, vol. 139, p. 109540, 2025.

[257] S. Malakar, S. Paul, S. Kundu, S. Bhowmik, R. Sarkar, and M. Nasipuri, "Handwritten word recognition using lottery ticket hypothesis based pruned cnn model: a new benchmark on cmaterdb2. 1.2," *Neural Computing and Applications*, vol. 32, pp. 15209–15220, 2020.

[258] C. Schuldt, I. Laptev, and B. Caputo, "Recognizing human actions: a local svm approach," in *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.*, vol. 3, pp. 32–36, IEEE, 2004.

Appendix

A. Time Complexity of GWO-VS

The time complexity of the proposed GWO-VS is mostly dependent on the parameters: (i) population size of GWO (say, P_G) and the number of candidate solutions of VS (say, P_V), (ii) classifier's training (say, Tr) and testing (say, Te) time, (iii) feature dimension (say, D), and (iv) the maximum number of iterations till which the GWO algorithm (i.e., $iter_{max}$ in Figure 4.12) and the VS algorithm (i.e., j_{max} in Figure 4.12) are executed to find the near optimal solution. It is to be noted that Tr and Te are directly proportional to D and the number of samples. In the GWO-VS algorithm, the VS algorithm runs once in each iteration of the GWO algorithm. Let us consider the time complexity of the GWO and VS algorithms are $O(G)$ and $O(V)$ respectively for one iteration. Now, the $O(G)$ can be calculated as the sum of the time taken by the following steps of the GWO algorithm.

- Initialization of candidate solutions takes $O(P_G \cdot D)$ time.
- The fitness calculation can be performed in $O(P_G \cdot (Tr + Te))$ time. Since $Tr \gg Te$. Since $Tr \gg Te$, this step takes $O(P_G \cdot Tr)$ time.
- The ranking of the candidate solutions is performed using $O(P_G \log(P_G))$ time.
- The position updating phase in the GWO part can be done in $O(P_G \cdot D)$ time.

Hence, $O(G) = O(P_G \cdot D) + O(P_G \cdot Tr) + O(P_G \log(P_G)) + O(P_G \cdot D)$. The $O(V)$ can be calculated as the sum of the time taken by the following steps of VS algorithm.

- Generation of initial candidate solutions requires $O(P_V \cdot D \cdot \mathcal{G})$ time, where $\mathcal{G}$ is the time taken by multivariate Gaussian distribution function.
- The fitness calculation can be performed in $O(P_V \cdot (Tr + Te))$ time. Like the GWO case, here also the step can be performed using $O(P_V \cdot Tr)$ time.
- The ranking of the candidate solutions is performed using $O(P_V \log(P_V))$ time.
- Substitution of a candidate solution of GWO with a better candidate solution obtained by employing VS algorithm can be performed in $O(P_V \cdot D)$ time.
- The search space reducing process takes $O(P_V \cdot \mathcal{G})$ time.

Hence, $O(V) = O(P_V \cdot D \cdot \mathcal{G}) + O(P_V \cdot Tr) + O(P_V \log(P_V)) + O(P_V \cdot D) + O(P_V \cdot \mathcal{G})$. The total time complexity of VS algorithm is $j_{max} \cdot O(V)$. This phase is run for each

candidate solution of GWO approach. Hence, the total time is taken by the GWO-VS algorithm (say, T) to generate a better solution is

$$T = O(G) \cdot O(V) \cdot j_{max} \cdot iter_{max}$$

i.e.,

$$T = \big(O(P_G \cdot D) + O(P_G \cdot Tr) + O(P_G \log(P_G)) + O(P_G \cdot D)\big) \cdot \big(O(P_V \cdot D \cdot \mathcal{G}) + O(P_V \cdot Tr) + O(P_V \log(P_V)) + O(P_V \cdot D) + O(P_V \cdot \mathcal{G})\big) \cdot j_{\max} \cdot \text{iter}_{\max}$$

However, for a given problem, the number of training samples and classifier is predefined i.e., in this case, Tr takes almost constant time. Hence, Tr can be treated as constant. Similarly, D and $\mathcal{G}$ can be treated as constant. Hence, T can be written as

$$T = (O(P_G) + O(P_G) + O(P_G \log(P_G)) + O(P_G)) \cdot (O(P_V) + O(P_V) + O(P_V \log(P_V)) + O(P_V) + O(P_V)) \cdot j_{max} \cdot iter_{max}$$

Or, $T = (3 \cdot O(P_G) + O(P_G \log(P_G))) \cdot (4 \cdot O(P_V) + O(P_V \log(P_V))) \cdot j_{max} \cdot iter_{max}$
Or, $T = O(P_G \log(P_G)) \cdot O(P_V \log(P_V)) \cdot j_{max} \cdot iter_{max} \approx O(P_G \cdot P_V \cdot j_{max} \cdot iter_{max})$.

B. GWO-VS Algorithm

Algorithm 6.1: GWO-VS Algorithm

Input: Population size N, maximum iterations $iter_{\max}$
Output: Optimal individual position W_α and best fitness values
Randomly initialize the positions of N individuals to form the population;
Initialize parameters a, A, k, and set $iter \leftarrow 0$;
while $iter < iter_{\max}$ *or stopping criteria not met* **do**
    Calculate the fitness value of each individual and update W_α, W_β, and W_δ;
    Update positions of individuals using Equation 4.9;
    Initialize $j \leftarrow 0$;
    while $j < j_{\max}$ **do**
        Generate modified candidate solutions using Equation 4.20;
        Choose the best agent from the solution set to replace the agent;
        Decrease the search radius using Equation 4.16 and Equation 4.17;
        $j \leftarrow j + 1$;
    end
    Update a, A, and k;
    $iter \leftarrow iter + 1$;
end
return *Optimal individual position* W_α *and best fitness values*;

Neural Computing and Applications (2023) 35:9363–9380
https://doi.org/10.1007/s00521-023-08201-z

S.I.: LATIN AMERICAN COMPUTATIONAL INTELLIGENCE

A hierarchical feature selection strategy for deepfake video detection

Sk Mohiuddin[1] · Khalid Hassan Sheikh[2] · Samir Malakar[1] · Juan D. Velásquez[3,4] · Ram Sarkar[2]

Received: 28 March 2022 / Accepted: 6 January 2023 / Published online: 6 February 2023

Abstract
Digital face manipulation has become a concern in the last few years due to its harmful impacts on society. It is especially concerning for high-profile celebrities because their identities can be easily manipulated using mobile or web applications such as FaceSwap and FaceApp. These manipulated faces are so close to real ones that it becomes really hard to detect them, even with bare eyes. Though deep learning-based models are predominantly used by researchers, they hardly check for the presence of irrelevant or redundant features produced by those models. To this end, we have proposed a hierarchical feature selection (HFS)-based method to detect deepfake images or videos. First, we have extracted both handcrafted and deep learning features from the inputs. Next, the HFS is applied to select a near optimal set of features. In each stage of it, a hybrid feature selection method is employed that integrates a population-based meta-heuristic model, called Grey Wolf Optimization, and a single solution-based meta-heuristic model, called the Vortex Search algorithm. We have evaluated our model on three publicly available datasets, namely Celeb-DF (V2), FaceForensics++, and Deepfake Detection Challenge (DFDC). The model provides 99.35%, 99.16%, and 85.67% AUC scores on the Celeb-DF (V2), FaceForensics++, and DFDC datasets, respectively, while using only 11.50%, 12.65%, and 10.22% of actual features. Besides, our model outperforms most of the state-of-the-art methods found in our literature review and evaluated on these three datasets.

Keywords Deepfake detection · Handcrafted features · Deep learning features · Hierarchical feature selection

1 Introduction

Digital face manipulation has become a real concern in social media as well as in our society due to its detrimental consequences. Although face manipulation, in its early stages, used to be done for comic purposes, later it became a cause for concern for celebrities and political figures. Also, in the past, due to technological limitations, it was very difficult to create realistic face images, and thus identification of the manipulations could be done with ease with bare eyes. However, recent developments in artificial intelligence (AI) based on deep neural networks, especially generative adversarial networks (GANs), let forgers alter the identity of a person with almost no perceivable signs. As a result, existing faces or face attributes such as eyes and mouth in a video or image are very intelligently replaced or altered by borrowing the corresponding parts from other subjects. Such a manipulated approach is known as deepfake, which generates very natural face images, and thus it becomes really challenging to detect the changes made by bare human eyes. Figure 1 depicts how a target face image is coupled with a source image to generate a fake face image. Therefore, designing a solution to the

✉ Juan D. Velásquez
jvelasqu@dii.uchile.cl

Sk Mohiuddin
myselfmohiuddin@gmail.com

Khalid Hassan Sheikh
skkhalidhassan7@gmail.com

Samir Malakar
malakarsamir@gmail.com

Ram Sarkar
ram.sarkar@jadavpuruniversity.in

[1] Department of Computer Science, Asutosh College, Kolkata, India

[2] Department of Computer Science and Engineering, Jadavpur University, Kolkata, India

[3] Department of Industrial Engineering, University of Chile, Santiago, Chile

[4] Instituto Sistemas Complejos de Ingeniería (ISCI), Santiago, Chile

Fig. 1 Some examples of how target images/videos are used to convert source images/videos into realistic fake faces

problem of identifying manipulated faces from real ones becomes very critical.

In the past, researchers tried to detect GAN-based forged faces by learning straightforward properties of images like abnormal changes within faces [1], and left-over convolutional traces in the spatial domain [2] made by GAN. Extending the same in the temporal domain, object behaviour, such as eye blinking patterns [3], was tracked for detecting deepfake videos. These methods performed well when the underlying deepfaking methods leave some explicit visual clues. However, with the advent of Convolutional Neural Network (CNN)-based deep learning models and the presence of voluminous training data, current GAN models can generate fake face images with ease which possess almost no visual clues (see Fig. 1). GAN-aided tools like FaceApp[1] and FaceSwap[2] let people manipulate face images in videos just by pressing a few buttons on their digital devices. As a result, we have observed most of these videos/images on several social media platforms now and then. Hence, it becomes a pressing need to restrict the spreading of such manipulated digital content. Most researchers from every corner of the world are trying to come up with a suitable solution to overcome this problem.

To date, several methods [4–6] have been developed to detect deepfake videos or images using different CNN architectures. These methods employed some basic CNN models to extract the desired features. The main limitation of using such a simple CNN model is that it sees the face differently than humans do when detecting forgery. Particularly, the vanilla CNN-based fake face detectors look for left-over artifacts from a limited facial region, as shown in Fig. 2, while humans usually seek these from the entire face image. In Fig. 2, we have shown heatmaps and grad-CAM images of fake face images that are generated by fine-tuned XceptionNet [7] that is considered as the baseline model for most computer vision tasks.

To mimic human observation, most researchers [9–14] used several attention networks that can widen the search space observed by the vanilla CNN models. Although an attention mechanism helps researchers to replicate human understanding quite satisfactorily, it fails to perform a multi-view analysis of an object, i.e., checking the consistencies among different parts. Moreover, these attention models mostly employed convolution operations on the entire image to obtain the required attention map, which may highlight some unimportant regions. Besides, they demand more computational capacity than a vanilla CNN model and might overfit due to a lack of training data.

Apart from these typical issues related to deep learning models, another important issue that may arise is that the feature set used by the existing models may contain some irrelevant and/or unimportant features [15]. The presence of irrelevant features does not add any additional knowledge to the classifiers, but rather sometimes misleads the classifiers while performing classification tasks [16, 17]. As a result, the underlying classifier not only fails to generate the desired output but also takes a considerably longer training time [16, 18]. The use of a good feature selection (FS) method would help overcome the aforementioned limitation through eliminating non-important features.

In this paper, we have designed a deepfake video detection method that delves into the above research arguments. The proposed method combines hand-crafted features with deep learning features. Hand-crafted features are designed to check similarity among different parts of a face image, while deep learning features represent root level purity. For deep feature extraction, we have used fine-tuned XceptionNet [7] considering its success in face manipulation detection [4]. In the case of hand-crafted features, we have first extracted Daisy features [19] from image patches owing to their usefulness for several image classification problems [20]. We have then estimated the feature level similarity score among the patch-level Daisy features to mimic the multi-view analysis used in many pattern recognition problems [19, 21]. Next, to select a near optimal feature set, we have employed a hierarchical feature selection (HFS) technique considering its strength in selecting a better feature subset than using an FS at one go while diversified features are combined [22]. As a base FS model, we have designed a new hybrid FS technique that integrates a population-based meta-heuristic, called Grey Wolf Optimizer (GWO) [23], with a single solution-based FS technique, called vortex search (VS) algorithm [24] which have been termed hereafter as GWO–VS algorithm. We have fed only the selected final feature set into a two level multi-layer perceptron (MLP) to distinguish real images/videos from their fake counterparts.

[1] https://apps.apple.com/gb/app/faceapp-ai-face-editor/id1180884341.

[2] https://github.com/MarekKowalski/FaceSwap.

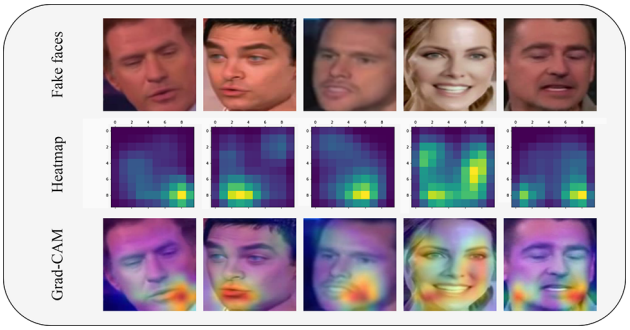


Fig. 2 Heatmap and Grad-CAM visualization [8] of some fake faces generated using fine-tuned XceptionNet model

Our main contributions are as follows:

- The development of a deepfake image/video detection technique that couples deep learning features and handcrafted features that represent similarity among different patches in an image using the HFS technique.
- The choice of an HFS technique for deepfake video or image detection over traditional non-HFS techniques for selecting a feature subset from a diverse set of features. To the best of our knowledge, this is the first time that an HFS technique has been used in this application.
- The combined use of GWO and VS methods (i.e., the GWO–VS algorithm) enhances the GWO algorithm's exploitation capacity, and thus, it is used as the fundamental FS scheme in the HFS technique.
- The evaluation of the HFS model on three publicly available datasets and its comparison with state-of-the-art methods. The proposed HFS model not only reduces more than 85% of features present in the original feature set but also improves the classification performance.

The remaining part of this article is organized as follows. Section 2 describes some previous methods. The proposed method is described in Sect. 3 while Sect. 4 is used for results and discussion. Finally, the paper is concluded in Sect. 5.

2 Related work

2.1 Deepfake detection techniques

A few works used handcrafted features. For example, McCloskey and Albright [1] analyzed colour level discrepancies between GAN generated manipulated face images and the corresponding actual camera captured images to design a face forensic method. Durall et al. [2] used Discrete Fourier Transform (DFT) to convert facial spatial information into a 1D power spectrum, which was then fed to classifiers like the logistic regression classifier and the support vector machine (SVM) to distinguish fake videos from real ones. Despite their simplicity, these methods performed well when all three components: the source image, the target image, and the underlying faking algorithm were previously known. However, in reality, such information might not always be available, and even the modern synthesis image/video preparation methods do not expose the difference between real and synthesized images/videos that much. In such cases, the performance of these detection methods degrades. As a result, several methods tracked the discrepancies in visual artifacts in the temporal domain among the facial attributes like eyes, teeth, and the head position. Matern et al. [25], for example, used separate attributes like iris region, and teeth color to extract color features. This method performed well for the videos that have only frontal faces in most of the frames and the facial attribute under consideration is well identifiable. But all the videos do not possess such features and thus non-availability of certain attributes may lead to a false prediction. Yang et al. [26] extracted 68 facial landmarks from central face regions to estimate head poses. The hypothesis that they considered is the position of facial landmarks is shifted during tampering. However, their method is limited to detecting forgery whenever tampering takes place in the region outside of the landmark area.

In recent times, researchers mostly relied on deep learning architectures to come up with a feasible solution. Tolosana et al. [27] used the concept of transfer learning where they considered both the full-face and different face components like eyes, and nose as inputs to their model. Pre-trained Xception model and Capsule Network were used as the backbone CNN model. Rossler et al. [28] experimented exhaustively using several CNN-based deepfake detection methods on a self-made dataset which was later made public to the research community. They showed that XceptionNet [7] performed the best among others while trained and tested on different datasets setups. To focus on mesoscopic information in an image, a variant of the Inception module was used by Afchar et al. [29]. Two different CNN architectures viz., Meso-4 and MesoInception-4 were proposed by them to accomplish the forgery detection task. Amerini et al. [30] used the unusual motion of facial components for deepfake detection. In this work, a CNN-based model, called PWC-Net was used to extract optical flow matrices from the face images, which were then fed into VGG16-based semi-trainable model to filter out authentic videos.

In contrast to vanilla CNN-based methods, several works [10–13, 31] suggested integrating attention mechanism with the CNN model to focus on wider regions of the facial image. For example, Wang and Deng [10] detected

face forgeries by an attention-based method that guided the CNN aided detector to focus on the wider facial region for better detection. In this work, the authors utilized augmented data for better training of their designed attention model. Dang et al. [31] introduced another attention model to concentrate on the manipulated facial regions while extracting deep learning features. To exploit the local feature of the face region, an attention mechanism was applied by Chen et al. [11] that fused information presented in RGB and frequency domain that extends the similarity pattern of the regions. Zhao et al. [13] employed multiple attention models for deepfake detection while Miao et al. [12] utilized two attention models: one that guides the model to learn the forgery region (FR) and the other that enforces the model to pay less attention to identity semantics using a landmark guided dropout module (LM). They called their model FRLM. However, using attention mechanisms with CNN makes the models computationally expensive and may occasionally result in model overfitting. From a technical point of view, these methods tried to generate better feature maps, however, they did not investigate the presence of noisy or irrelevant features that might affect the end performance of the model. Apart from these, recently, two surveys [32, 33] reviewed various deepfake detection techniques.

2.2 FS techniques

In recent times, meta-heuristic algorithm-based FS techniques have gained enormous attention in the research community due to their wider applicability in diversified fields of computation and beyond. Three broad categories of FS methods [17, 34], namely filter, wrapper, and embedded are found in the literature. The main idea behind a filter-based FS approach is to rank features without using a classifier. Methods following filter-based approaches, in general, rely on statistical characteristics like correlation scores, and mutual information among the features to rank the features [17]. They are not only simple to implement but also computationally inexpensive. However, the main problem with the filter-based FS methods is that most filter methods evaluate features one by one without considering feature subsets.

On the contrary, a wrapper-based FS method first selects some subsets of features, known as candidate solutions, from the original feature set through different predefined norms and then evaluates the merits of the selected features based on an objective function. This is an NP-hard problem because determining the best feature subset from a set of N features has a complexity of $2N$ [17, 35]. As a result, the researchers employ a variety of meta-heuristic methods to identify the potentially near-optimal feature subset [16, 35, 36]. Though these methods are computationally expensive compared to filter methods, in general, they outperform the filter methods. An embedded method is aimed at reducing the computational time taken by wrapper-based methods by coupling the advantageous aspects of both filter and wrapper methods. In this case, the FS method is an integral part of a classification model. These methods, in general, fail to generate a low dimensional feature subset and are available with the classification methods. In short, each type of FS method has some benefits and limitations [35, 36]. Nowadays, FS methods based on meta-heuristic algorithms are a popular choice due to their advantages over the filter and embedded approaches. Meta-heuristic methods mostly followed one of two broad categories: population based or single solution based.

- *Evolutionary algorithms* Darwinian evaluation process inspired algorithms are evolutionary algorithms. The most commonly known algorithm in this category is Genetic Algorithm (GA) [37]. GA tries to mimic the biological process of crossover and mutation of genes. Some other Evolutionary algorithms are Bio-geography-based optimizer [38], and Differential Evolution [39].
- *Swarm-based algorithms* This category of algorithms is inspired by the behaviour of swarms, herds, and flocks of various creatures of nature. They try to mimic the swarm's behaviour to find optimum solutions. One of the pioneers of swarm-based algorithms is the particle swarm optimization (PSO) algorithm [40]. It mimics the social behaviour of a flock of birds. Some of the popularly used swarm-based algorithms are GWO [23], cuckoo search [41], and artificial bee colony algorithm [42].
- *Physics-based algorithms* Physics-based meta-heuristic algorithms are inspired by physical phenomena of nature. One of the popularly used physics-based meta-heuristic algorithms is simulated annealing (SA) [43]. SA is inspired by the metallurgical process of annealing. It is a single solution-based meta-heuristic. Other popularly used similar algorithms are harmony search (HS) [44], and gravitational search algorithm (GSA) [45].

In the single solution-based approach, a candidate solution tries to explore its nearby solutions over the iterations. It reduces the search locality to find an optimum solution. Some of the single solution-based algorithms are hill climbing [46], Tabu search [47], and VS algorithm. Often population-based meta-heuristics fail to attain the near optimal solution. This limitation could be overcome by using a population-based method coupled with a single solution-based algorithm. By doing so one can increase the exploration capabilities of the base meta-heuristic algorithm by a significant margin. In [48], a hybrid meta-

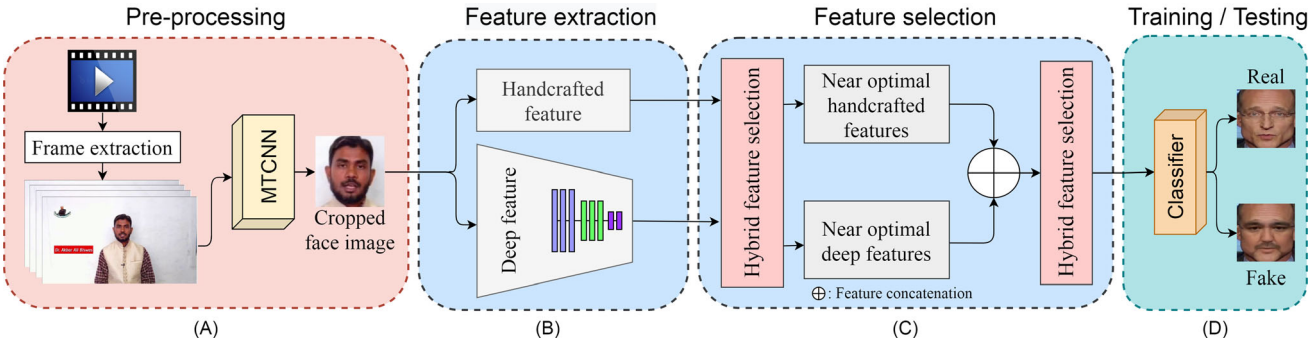


Fig. 3 Workflow of the proposed method having three stages **A** Preprocessing: it performs operations to generate cropped face image from the input video using the MTCNN-based face detection method, **B** Feature extraction: it extracts handcrafted and deep features from the face image, **C** Feature selection: it selects a near optimal feature set by employing the HFS technique, and **D** Training/Testing: it uses a classifier to classify input videos as forged or real

heuristic algorithm was proposed that used β-hill climbing to improve the exploration capability of the bat algorithm. In another work, Wang et al. [49] used the VS algorithm to overcome the problem of pre-mature convergence of the artificial bee colony optimization algorithm. Some more similar hybrid algorithms could be found in [34, 46, 49–51]. When different types of features are used for a classification problem, various authors employed an HFS approach in which the FS method was applied to combine multiple types of features at different levels. For example, the authors in [22] used various texture and shape descriptive features to recognize handwritten Bangla words.

3 Proposed work

In this work, we have proposed a method to distinguish face manipulated videos from real videos. At the outset, we have utilized I-frames of a questioned video (i.e., a test video) for further processing. The reason behind the consideration of I-frame over other categories of video frames is that it resides in uncompressed form and holds maximum color information compared to others. Hence, we have first extracted the I-frames from the questioned video and then passed it through the multi-task cascaded CNN (MTCNN) [52], a state-of-the-art face detection method, to extract the potential face region. Next, two types of features: handcrafted and deep learning features are extracted from these cropped face images. The features selected by employing the proposed HFS technique are fed to an MLP classifier for detection. In the HFS technique, we have used the GWO–VS technique. Figure 3 shows the steps followed in the proposed method.

3.1 Data processing

To detect the manipulated face in videos, I-frames are extracted using the ffmpeg[3] technique. Since face manipulations are performed on face regions, at the beginning, we have tried the OpenCV Haar Cascade method to detect and crop the face regions from the extracted frames. However, this detection method suffers from false detection in many cases. Therefore, we have used a CNN-based face detection algorithm, called MTCNN, which performs well for different video qualities with diverse face alignments and sizes. After detecting the bounding box of each face region, we have increased the size of the bounding box by 10 pixels in each direction to include some background information around the face region. Finally, the cropped face images are resized to (225×225) and (299×299) for handcrafted and deep learning feature extraction purposes, respectively.

3.2 Feature extraction

In this work, we have considered both handcrafted and deep learning features described below.

3.2.1 Handcrafted feature extraction

To look at the face region following a multi-view approach [21], the cropped faces have been partitioned into 25 equal-sized image patches having dimension 45×45. Next, we have extracted Daisy features [19] from each image patch. Daisy is a gradient orientation histogram-based image descriptor that describes both dense and shallow artifacts on the image patches. It is designed to extract dense properties of the images using a circularly symmetrical kernel and the Gaussian weight. The 'scikit-image' library is used to extract a bag of 25 Daisy descriptors each having

[3] https://ffmpeg.org/.

104 elements from a single image patch. This way a single patch is represented by a 2600-dimensional feature vector.

Next, we have estimated feature similarity using Euclidean distance (see Eq. 1) among all the extracted patches. In Eq. 1, E_{ij} represents the Euclidean distance between ith and jth patches while $\vec{P}_i$ and $\vec{P}_j$ denote feature vectors for the ith and jth patches, respectively. For every value of $i \in \{1, 2, \ldots, 24\}$, we have varied the value of j from $i+1$ to 25 (i.e., $j \in \{i+1, i+2, \ldots, 25\}$) to extract similarity among all patches. Finally, concatenating all E_{ij}s, a 300-dimensional feature vector is obtained from a single face image. Figure 4 depicts the feature extraction process pictorially.

$$E_{ij} = |\vec{P}_i - \vec{P}_j| = \sqrt{(P_i - P_{i+1})^2 + (P_i - P_{i+2})^2 \ldots + (P_i - P_{25})^2} \quad (1)$$

3.2.2 Deep feature extraction

The proposed method uses fine-tuned XceptionNet [7] network to learn the important features from face images. The architecture of the Xception model is based on depth-wise separable convolutional layers that are inspired by the Inception model. Each channel uses 1×1 depth-wise convolutions to capture the cross-channel correlations. During the depth-wise separable convolutions, the Xception network performs channel-wise convolutions first and then employs 1×1 convolution while the Inception network uses 1×1 convolution first and then uses channel-wise convolutions. In addition to this, the number of layers is reduced from 159 layers in Inception V3 to 126 layers in the Xception architecture. The fine-tuned Xception model extracts 2048-dimensional deep learning features from a face image of size 299×299.

3.3 Feature selection

In this work, we have proposed an HFS model to perform FS. A hybrid FS method is designed using the advantageous aspects of GWO [23] and VS algorithm [24]. Here we have described all the components of the proposed HFS model.

3.4 Grey wolf optimizer

GWO is inspired by the preying strategies of a pack of grey wolves. One of the interesting characteristics of grey wolves is that they form packs with a well-defined hierarchical structure. A typical pack of grey wolves consists of 5–12 grey wolves on average. The leader of the pack is known as the alpha who is the main decision maker. Instructions of the alpha wolf are followed by the remaining wolves of the pack. Despite the dominance of the alpha wolf in the decision making process, the pack also shows traits of democratic behaviour as the alpha wolf takes advice from others. The second most important type of wolf in the hierarchy is beta wolves who instruct other wolves in the pack on behalf of the alpha and provide feedback to the alpha. They are also the most suitable candidate for being alpha. The third most important type of wolf is the delta, and the lowest ranked wolves are called the omega. Delta has an upper hand over omega, although they submit to the command of alpha and beta. One of the other interesting factors of a grey wolf pack is their hunting behaviour. The main phases of their hunting are as follows:

- Tracking, chasing, and approaching the prey.
- Pursuing, encircling, and harassing the prey until it stops moving.
- Attacking the prey.

For FS, a solution can be represented in the following way $\vec{F} = [f_0, f_1, \ldots, f_{N-1}]$, where N and f_i $(i = 0, 1, \ldots, N-1)$ represent the number of features and ith feature in the feature set, respectively. Here, any f_i can take a value of 0 or 1 to indicate non-inclusion or inclusion of ith feature in the final feature subset. Suppose a position represented by any candidate solution at a particular iteration of a wolf at some instance is $\vec{W}$. The position of α, β, and δ wolves are $\vec{W}_\alpha$, $\vec{W}_\beta$, and $\vec{W}_\delta$, respectively. The mathematical model with which the wolves in the pack update their position is shown in Eqs. 2 and 3.

$$\vec{D}_\alpha = \vec{k1} \cdot \vec{W}_\alpha - \vec{W}, \vec{D}_\beta = \vec{k2} \cdot \vec{W}_\beta - \vec{W}, \vec{D}_\delta = \vec{k3} \cdot \vec{W}_\delta - \vec{W} \quad (2)$$

$$\vec{W}_1 = \vec{W}_\alpha - \vec{A_1} \cdot \vec{D}_\alpha, \vec{W}_2 = \vec{W}_\beta - \vec{A_2} \cdot \vec{D}_\beta, \vec{W}_3 = \vec{W}_\delta - \vec{A_3} \cdot \vec{D}_\delta \quad (3)$$

The parameters: A_i and k_i $(i \in \{1, 2, 3\})$ in Eqs. 2 and 3 are defined by Eq. 4.

$$A_i = 2 \cdot \vec{a} \cdot \vec{r_i^1} - \vec{a}, k_i = 2 \cdot \vec{r_i^2} \quad (4)$$

In Eq. 4, $\vec{r_i^1}$ and $\vec{r_i^2}$ are two random vectors and components of $\vec{a}$ are linearly decreased over the iteration from 2 to 0. It is to be noted here that if $\vec{X} = [X_0, X_1, \ldots, X_{N-1}]$ and $\vec{Y} = [Y_0, Y_1, \ldots, Y_{N-1}]$ are two candidate solutions or N-dimensional vectors then $\vec{X} \cdot \vec{Y}$ is defined by Eq. 5.

$$\vec{X} \cdot \vec{Y} = [X_0 \cdot Y_0, X_1 \cdot Y_1, \ldots, X_{N-1} \cdot Y_{N-1}] \quad (5)$$

Suppose $\vec{W}$ can be represented as $[w^1, w^2, \ldots, w^{N-1}]$. The possible position of the wolf in the next iteration is defined by the Eq. 6.

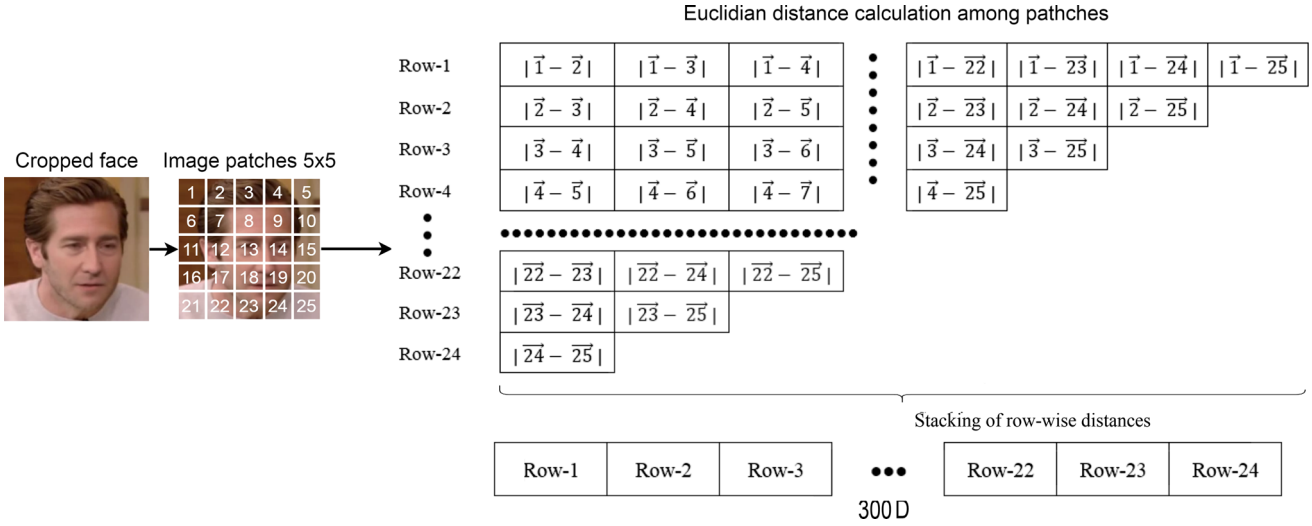


Fig. 4 Illustration of the hand-crafted feature extraction process from a cropped face image. The second image from the left shows the label number $t \in \{1, 2, \ldots, n\}$ of the extracted patches. Next, from each patch, we have extracted Daisy features represented by $\vec{t}$ which are then used to find similarities among the different patches. The pair-wise norm-2 distances (i.e., $|\vec{i} - \vec{j}|$, where $i \in \{1, 2, \ldots, n-1\}$ and $j \in \{i+1, i+2, \ldots, n\}$) are then calculated and stored in list named as Row $-\ i$. Thus, the length of Row $-\ i$ is $n - i$. Finally, all these lists are stacked to form the final feature vector. The length of feature vector is $= (n-1) + (n-2) + \cdots + 1 = \frac{n(n-1)}{2}$. In our case $n = 25$ and thus we have obtained a feature vector of length 300

$$\vec{W}(t+1) = Func\left(\frac{\vec{W}_1 + \vec{W}_2 + \vec{W}_3}{3}\right) \tag{6}$$

In Eq. 6, *Func*() is a transfer function that converts a continuous value into a discrete binary value. Next, the fitness values of the newly generated position of a wolf (i.e., the new candidate solution) and its old position are calculated. If the fitness value of the new position is better than the previous one, we have updated the position of the wolf.

3.5 Vortex search algorithm

VS algorithm was proposed by Doğan and Ölmez [24], which is a single solution-based meta-heuristic algorithm. Suppose $S(\vec{V})$ is the set of candidate solutions generated from the current solution $\vec{V}$. In the next phase i.e., in the replacement phase, a suitable solution (say, $\vec{V}' \in S(\vec{V})$) is selected from the set of generated candidate solutions that may replace the current solution. This process continues until the termination condition is met. The phases of the VS algorithm are as follows:

1. Generating the initial solution.
2. Generating the set of candidate solutions.
3. Ensuring all candidate solutions lie in the search space.
4. Replacing the current solution with the best candidate solution if any.
5. Decreasing the radius of search space.

An initial solution (say, $\vec{V}$) is determined by Eqs. 7 and 8.

$$\vec{V} = [V_0, V_1, \ldots, V_{N-1}] \tag{7}$$

$$V_i = \frac{u\text{Lim}_i + l\text{Lim}_i}{2} \tag{8}$$

In Eq. 8, $u\text{Lim}_i$ and $l\text{Lim}_i$ are the maximum and minimum possible values in the ith dimension, respectively. Please note that $i \in \{0, 1, 2, \ldots, N-1\}$ where N is the dimension of the single solution which is the cardinality of the feature set. Next, a set of neighbour solutions (say, $S(\vec{V})$) is generated randomly using the Gaussian distribution. The multivariate Gaussian distribution can be formulated using Eq. 9.

$$p(\vec{V}|\vec{x}, \Sigma) = \frac{1}{\sqrt{(2\pi)^N |\Sigma|}} \exp\left\{-\frac{1}{2}(\vec{x} - \vec{V})^{\mathrm{T}} \Sigma^{-1} (\vec{x} - \vec{V})\right\} \tag{9}$$

Here $\vec{x}$ is a randomly generated N dimensional vector. For spherical distribution, the Σ can be defined using Eq. 10.

$$\Sigma = \sigma^2 \cdot [I]_{N \times N} \tag{10}$$

Here $[I]_{N \times N}$ is an identity matrix of order $N \times N$, and σ^2 represents the variance of the distribution. The initial standard deviation is calculated using eq. 11.

$$\sigma_0 = \frac{\max(u\text{Lim}_0) - \min(l\text{Lim}_0)}{2} \tag{11}$$

σ_0 can be considered radius of distribution. The initial phases of this algorithm need weak locality for better exploration ability; hence the maximum possible radius of

distribution is considered here. Before the replacement of the center of distribution that is the current single solution, all candidate solutions that lie beyond the search space must be adjusted into the search space following Eq. 12.

$$v'_i = \begin{cases} v'_i, & \text{if } lLim_i \le v'_i \le uLim_i \\ \text{rand}(uLim_i - lLim_i) + lLim_i, & \text{else} \end{cases} \tag{12}$$

In Eq. 12, v' is any of the random candidate solutions. In the next step, if the best candidate solution outperforms the existing solution based on fitness score, the existing solution is modified to the best candidate solution. Afterward, this algorithm gradually decreases the radius of the search space by reducing the standard deviation (SD) following Eqs. 13 and 14.

$$\sigma_t = \sigma_0 \cdot gammaincinv(x, \alpha_t) \tag{13}$$

$$\alpha_t = \alpha_0 - \frac{\text{iter}}{\text{TotalIter}} \tag{14}$$

In Eq. 13, *gammaincinv*() is the incomplete inverse gamma function [53]. In summary, VS starts with a solution first and then generates a set of candidate solutions using the Gaussian distribution defined in Eq. 9 keeping the solution as the center. Next, using steps 3–5 of the VS algorithm mentioned earlier, the VS algorithm tries to find a better solution.

3.6 Proposed hybrid FS method

The GWO, a population-based meta-heuristic algorithm, suffers from premature convergence like other similar algorithms. Inspired by the work by Wang et al. [49], here we have used a single solution-based meta-heuristic, called VS algorithm to deal with premature convergence of GWO. In the proposed hybrid FS method (i.e., GWO–VS), the VS algorithm helps in finding a better position for the wolves at each iteration. In other words, the VS algorithm, used in the GWO–VS algorithm, starts with the position of a wolf i.e., a candidate solution (say, $\vec{W}_i$, where $i = 0, 1, \ldots, P_G - 1$, and P_G is the number of wolves in a pack) from the GWO algorithm as the initial solution and then follows the described steps to find a better position for the wolf under consideration. The entire process of the GWO–VS-based FS technique is described through Eqs. 15, 16, and 17.

$$\text{Set}(\vec{W}_i) = \{\vec{S}_0, \vec{S}_1, \ldots, \vec{S}_{P_V - 1}\} \tag{15}$$

In Eq. 15, P_V is the number of candidate solutions of VS algorithm that consider the ith candidate solution (i.e., $\vec{W}_i(t)$, where $i = 0, 1, \ldots, P_G - 1$) of the GWO algorithm as the initial solution. It is to be noted here that $\vec{S}_k, k = 0, 1, \ldots, P_V - 1$ is a vector in N-dimensional feature space.

$$\vec{W}_i(t+1) = \text{Best}(\text{Set}(\vec{W}_i(t))) \tag{16}$$

Equation 16 describes the selection of the best candidate solution for the $(t+1)$th iteration of VS algorithm. Here, $t = 1, 2, \ldots, j_{\max}$ and $j_{\max}$ is the number of iterations for which VS algorithm runs.

$$p(\vec{w}|\vec{x}, \Sigma) = \frac{1}{\sqrt{(2\pi)^N |\Sigma|}} \exp\left\{\frac{1}{2}(\vec{x} - \vec{w})^T \Sigma^{-1} (\vec{x} - \vec{w})\right\} \tag{17}$$

The VS algorithm uses a multivariate Gaussian distribution (see Eq. 17) to generate candidate solutions. Here w is the current solution or center of the search space, and x is a random N dimensional vector. Σ is the covariance matrix. The value of Σ can be calculated using Eqs. 10 and 11. Over several iterations, the scope of search space is similarly reduced using Eqs. 13 and 14. The flowchart of the GWO–VS algorithm is shown in Fig. 5. The time complexity of the GWO–VS algorithm is approximately $O(P_G{*}P_V{*}\text{iter}_{\max}{*}j_{\max})$, where P_G, P_V, $\text{iter}_{\max}$, $j_{\max}$ represent population size of GWO, the population size of VS, the maximum iteration number of GWO, and the maximum iteration number of VS, respectively. It is to be noted here that whenever we have used the operator '*' it means the arithmetic multiplication operator. A detailed complexity analysis of the GWO–VS algorithm is provided in Section A of the supplementary document. We have also added the pseudocode of this algorithm in Section B of the supplementary document.

3.6.1 Fitness function

A fitness function is used to evaluate the performance of a candidate solution. In our work, the fitness function is designed to deal with a trade-off between the number of selected features (N_{selected}) and the classification accuracy (ACC), which is defined in Eq. 18. In this work, ACC is evaluated on a validation dataset and an MLP is used as a classifier.

$$fitness(\vec{w}) = wt * \text{ACC} + \frac{N_{\text{selected}}}{N} * (1 - wt) \tag{18}$$

In Eq. 18, $wt \in [0, 1]$ (in our case, $wt = 0.90$) and $(1 - wt)$ represent the weights for the recognition accuracy and the number of selected features, respectively.

3.6.2 Transfer function

As FS is a binary optimization problem, the solution vector can only contain '0' or '1', where '0' and '1' mean the exclusion and inclusion of the corresponding feature into

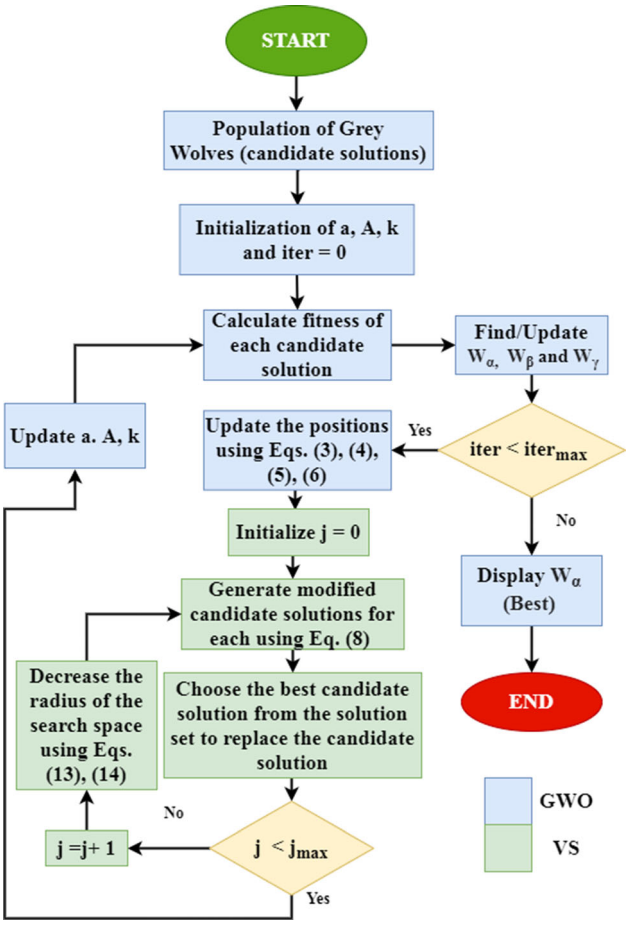


Fig. 5 Flowchart of the proposed hybrid FS technique

the selected feature set from the actual feature set, respectively. However, for GWO the value cannot be binary due to the nature of mathematical operations. To ensure that the output always stays within the expected range, we need to apply a transfer function on each candidate solution. This task is performed by the sigmoid (S-shaped) transfer function. The S-shaped transfer function is formulated using Eq. 19.

$$T(x) = \frac{1}{1 + e^{-x}} \tag{19}$$

$$X^d(t) = \begin{cases} 1, & \text{if rnd} < T(X^d(t)) \\ 0, & \text{if rnd} \geq T(X^d(t)) \end{cases} \tag{20}$$

In Eq. 20, $\text{rnd} \in (0, 1)$ is a random number.

3.7 Hierarchical feature selection

In the HFS model, multiple types of features are combined to come up with a near optimal feature subset to perform the said classification task. The combination and selection of features are carried out hierarchically. In the HFS model, firstly we have employed our proposed hybrid FS technique on

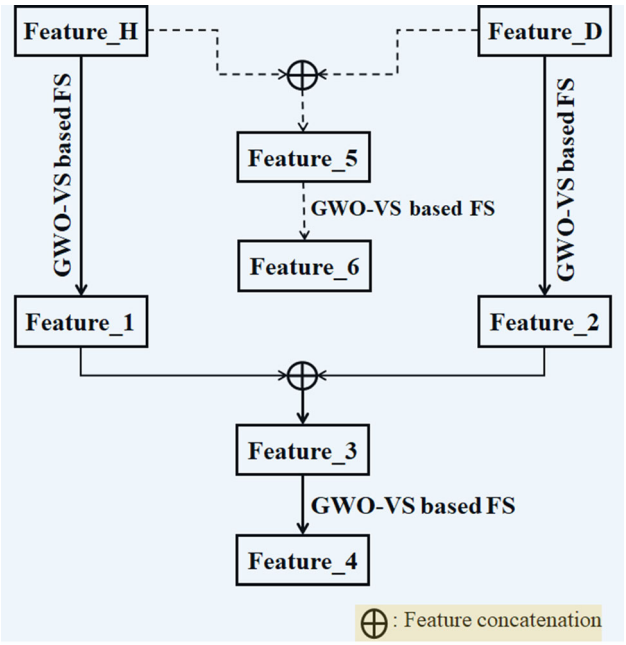


Fig. 6 In the proposed HFS model, steps followed to generate the feature vectors are connected using solid-lined arrows, while steps for the non-HFS stages are connected using dashed-lined arrows. Feature_4 and Feature_6 are the obtained near optimal features following HFS and non-HFS approaches, respectively

extracted handcrafted (Feature_H in Fig. 6) and deep learning features (Feature_D in Fig. 6) individually. This is done to remove the redundant features from each type of feature and we have obtained the near optimal feature sets: Feature_1 and Feature_2 from Feature_H and Feature_D, respectively, as shown in Fig. 6. Next, we have concatenated the obtained near optimal feature subsets and obtain a new feature subset (Feature_3 in Fig. 6). This combined feature set (i.e., Feature_3) might possess some irrelevant features [22]. Thus, at this end, the proposed hybrid FS model is again employed on the combined feature set and we have obtained the final feature subset (i.e., Feature_4 in Fig. 6). This feature set is considered for classification. Figure 6 shows the commonly used FS strategy, where features are concatenated first and then the combined features (i.e., Feature_5 in Fig. 6) are passed through the FS method to generate the near optimal features (Feature_6 in Fig. 6).

4 Results and discussion

In this work, we have designed an HFS technique to detect whether a questioned video/image has been manipulated by some deepfake techniques or not. The proposed model is evaluated on three popular video datasets—Celeb-DF (V2) [4] termed hereafter as CeDF, FaceForensics++ (FF++) [28] and Deepfake Detection Challenge (DFDC) [54].

Table 1 Distribution of videos and extracted frames/cropped faces used in training, validation, and test sets for both datasets

Dataset	# Videos in						# Frames/cropped faces in					
	Train set		Validation set		Test set		Train set		Validation set		Test set	
	Real	Fake	Real	Fake	Real	Fake	Real	Fake	Real	Fake	Real	Fake
CeDF	612	4399	100	900	178	340	1130	8022	100	900	178	340
FF++	700	700	200	200	100	100	2994	2876	200	200	100	100

4.1 Dataset preparation

To assess the performance of the proposed deepfake detection technique, we have used three prevalent benchmark datasets: CeDF [4], FF++ [28], and DFDC [54]. The CeDF contains 5639 high quality synthesis videos of different celebrities. All the synthesized videos are generated from 590 original videos collected from YouTube with different subjects like ages, genders, and backgrounds. It provides 518 videos (178 real and 340 fake) for assessing the performance of a new model. The FF++ video dataset contains 4 types of forged videos, namely Deepfakes, Face2Face, FaceSwap, and NeuralTextures. Each category contains 1000 videos generated from 1000 pristine videos. The whole video dataset is available in two different compression rates but here we have used the c23 version. Since the proposed method targets detecting deepfake videos thus in our experiments, we have considered only Deepfake category videos as fake videos and pristine category videos as real videos. At the outset of data preparation, we have divided the datasets into the train, test, and validation sets. Table 1 shows the breakup details for both datasets. The Sect. 4.8 presents details about dataset division and performance on the DFDC dataset.

Next, we have extracted frames from the train, test, and validation video sets. In the case of training video sets, equidistant frames in the temporal domain are chosen while for test and validation sets we have considered only the first I-frame from each video. Such choice of the first I-frame is inspired by the methods reported in [6, 14]. During the generation of cropped face images using MTCNN, we have considered only one face having the highest confidence score from each of the extracted frames. Table 1 shows the distribution frames/cropped face images extracted from both the video datasets.

4.2 Experimental setup

The Daisy features are extracted using 'scikit-image' library where the parameters are set as: radius = 13, rings = 2, histograms = 6 and orientations = 8. For fine-tuning XceptionNet, we have set learning rate = 0.0001, batch size = 32, number of output layers = 2, number of epochs = 50, steps per epoch = 40 and optimizer = Adam. For the implementation of the GWO–VS model, we have used the parameter values as suggested in [23] and [24] for the GWO algorithm and VS algorithm, respectively. However, for other meta-heuristic algorithms used here for comparison, we have considered the parameter values as used in [46].

4.3 Experimental results

In this section, we have described the performance of the proposed HFS-based deepfake detection method. The performance evaluation of the proposed HFS method considers three parameters namely, Area Under the ROC Curve (AUC) score, test accuracy, and the number of selected features. We have executed the proposed HFS model 10 times on both datasets to test the effect of the randomness of meta-heuristic results. The results are shown in Fig. 7. From the results is clear that in most of the cases, we have obtained 98.00% and 97.30% test accuracy on FF++ and CeDF datasets, respectively. The detailed experimental outcomes are shown in Tables 2 and 3 for CeDF and FF++ datasets, respectively. These results help us to understand the performance of the entire model as well as all its components (see Fig. 6). We can observe from Table 2 that test accuracy and AUC score obtained on CeDF using Feature_4 (obtained with the proposed HFS method) are 97.30% and 99.35%, respectively, which are the best performances among all the other features. Likewise, the HFS model provides the best result in terms of both test accuracy and AUC score on FF++ dataset (see Table 3). In this case, the test accuracy and AUC score are 98.00% and 99.16%, respectively. By closely analyzing the results in Tables 2 and 3, the following statements can be inferred:

- If we have compared the performance of Feature_D and Feature_H then it is found that Feature_D is dominating. However, by concatenating these two (i.e., Feature_5 in Fig. 6), an improved performance is obtained. From these results, we can safely comment that the use of handcrafted features representing multi-view analysis along with deep learning features is beneficial.
- To test the usefulness of Feature_H in solving the current problem we have performed another set of experiments with the proposed model by using the deep

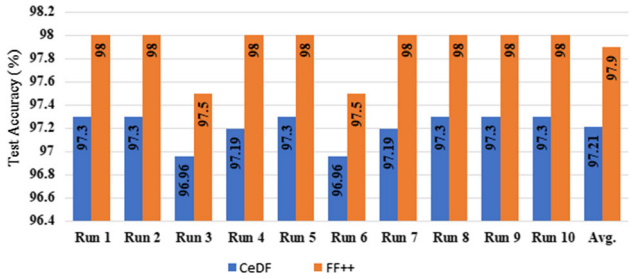


Fig. 7 Test accuracies of the proposed HFS-based deepfake detection method over ten independent runs

learning features extracted from MesoNet [29]. The results are shown in Table C.1 of Section C of the supplementary document confirming the usefulness of Feature_H in the context of the current problem.

- Almost in all the cases i.e., whenever the proposed hybrid FS (i.e., GWO–VS) is employed, we have observe improved performance and reduced feature dimension. This proves the usefulness of the proposed hybrid FS technique.
- Most of the methods, irrespective of the applications where they were used, generated the final feature vector by concatenating the features i.e., Feature_5 in Fig. 6 to perform the classification task. On both datasets, the use of the HFS to perform FS aided classification proves to be beneficial over this traditional approach. Proposed HFS not only reduces the feature dimension by 88.50%

Table 2 Performance of the proposed HFS model and its components (see Fig. 6) on CeDF dataset

Features	Performance metrics											
	Test accuracy (in %)				AUC score (in %)				# Selected features			
	Max	Min	Avg.	SD	Max	Min	Avg.	SD	Max	Min	Avg.	SD
Feature_H	65.64	64.76	65.16	0.299	73.49	73.07	73.32	0.157	300	300	300	0.00
Feature_D	95.37	94.68	95.01	0.249	98.88	97.92	98.51	0.394	2048	2048	2048	0.00
Feature_1	65.83	65.02	65.56	0.253	71.60	70.95	71.29	0.252	91	76	81.67	4.99
Feature_2	95.14	94.78	95.05	0.129	99.02	98.27	98.80	0.271	722	703	712	6.40
Feature_3	96.91	96.33	96.65	0.211	98.14	97.60	97.96	0.180	813	780	795.5	11.43
Feature_4	97.30	96.83	97.19	0.181	99.35	98.26	98.92	0.405	270	240	257	10.55
Feature_5	96.72	95.75	96.38	0.343	98.76	98.00	98.38	0.310	2348	2348	2348	0.00
Feature_6	96.91	95.95	96.45	0.374	98.14	98.10	98.21	0.136	830	790	812.33	14.53

Here, performances under Feature_4 and Feature_6 represent performances of HFS and non-HFS-based FS, respectively. Here Max, Min, Avg, and SD stand for maximum, minimum, average, and standard deviation, respectively

Table 3 Performance of the proposed HFS model and its components (see Fig. 6) on the FF++ dataset

Features	Performance metrics											
	Test accuracy (in %)				AUC score (in %)				# Selected features			
	Max	Min	Avg.	SD	Max	Min	Avg.	SD	Max	Min	Avg.	SD
Feature_H	63.5	62.5	62.90	0.374	70.22	69.84	70.04	0.148	300	300	300	0.00
Feature_D	96.00	95.50	95.70	0.245	98.34	97.8	98.13	0.211	2048	2048	2048	0.00
Feature_1	71.00	70.00	70.55	0.350	76.22	75.56	75.96	0.264	97	90	95	2.89
Feature_2	96.50	95.50	96.10	0.374	97.83	96.78	97.48	0.361	659	640	643.33	11.47
Feature_3	97.50	96.50	97.25	0.335	98.68	98.10	98.46	0.181	756	720	737.17	14.63
Feature_4	98.00	97.50	97.70	0.245	99.16	98.33	98.89	0.273	247	219	234	10.52
Feature_5	97.00	96.00	96.55	0.350	98.61	97.75	98.24	0.340	2348	2348	2348	0.00
Feature_6	97.00	96.50	96.80	0.245	98.68	98.20	98.44	0.162	714	690	703.83	9.39

Here, performances under Feature_4 and Feature_6 represent performances of HFS and non-HFS-based FS, respectively. Here Max, Min, Avg, and SD stand for maximum, minimum, average, and standard deviation, respectively

and 87.35% of the actual feature dimension for CeDF and FF++ datasets, respectively, but also improves the end performance (improvements on CeDF: test accuracy =0.58% and AUC score =0.59% and FF++: test accuracy=1.00% and AUC score =0.55%) over the traditional approach.

- In general, the methods that used FS to perform some classification tasks followed a non-HFS approach i.e., the authors generated Feature_6 as shown in Fig. 6. On both datasets, the use of the HFS to perform the FS task proves to be beneficial over this non-HFS FS approach. Proposed HFS not only selects 3.07 and 2.71 times fewer features to the dimension of Feature_6 for CeDF and FF++ datasets, respectively, but also improves the end performance (improvements on CeDF: test accuracy=0.39% and AUC score=0.36% and FF++: test accuracy=1.00% and AUC score=0.48%) over the single-stage FS approach.
- If we compare the performances obtained using Feature_3 and Feature_4 (see Fig. 6) then we can observe very little performance improvement for both datasets. However, a notable reduction in feature dimension can be observed. The dimension of Feature_4 is 3.01 and 3.06 times less concerning Feature_3 for CeDF and FF++ datasets, respectively. From these results, we can safely comment that the use of the proposed hybrid FS on Feature_3 is also beneficial while considering the dimension of the final feature set and the training cost of the model (see Fig. 8). Figure 8 also shows that the model with reduced features is trained almost 2.5 times faster than the one.

4.3.1 Statistical test among different near optimal features

We have also performed the statistical significance test to ensure the truthiness of the aforementioned observations. The objective of this statistical test is to test whether there is enough evidence to reject the null hypothesis: the performance of a near optimal feature subset (i.e., Feature_i, where $i = \{1, 2, 3, 5, 6\}$) over 10 independent trials is not significantly different from the performance of Feature_4 while performing deepfake detection. To do this, we have computed the p value using the Wilcoxon rank-sum test following the approach used in [55]. We reject the null hypothesis if the estimated result is less than 0.05 (i.e., $<$ 5% of the total). We have executed the FS stages/techniques that generate near optimal feature subsets: Feature_1, Feature_2, ..., Feature_6 10 times independently and recorded the performances in terms of AUC score, test accuracy, and the number of selected features. Based upon these records, we have estimated the p value, and the estimated p values are listed in Table 4. From the results recorded in Table 4, we can reject the null hypothesis at a 5% confidence level. Therefore, we can safely claim that the performance of the deepfake detection technique using Feature_4 is better than the performance of other cases i.e., performing deepfake detection using other selected near optimal feature subsets (i.e., Feature_i, where $i = \{1, 2, 3, 5, 6\}$) at the 5% significance level.

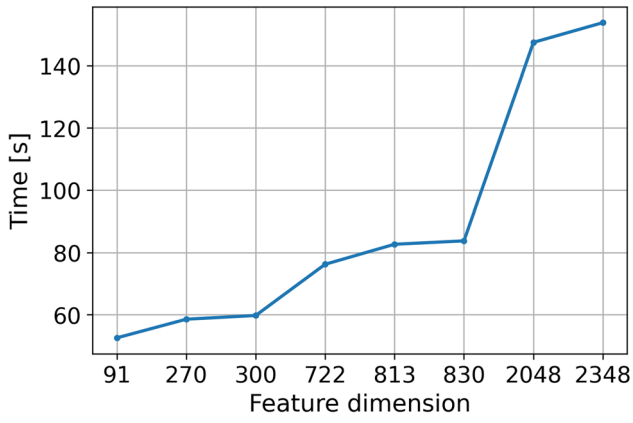


Fig. 8 Training time versus number of features for the CeDF dataset

4.4 Generalization capability of selected features

In this section, we have performed another set of experiments to estimate the generalization capability of the proposed HFS model. For this set of experiments, first, we have employed the HFS technique on the first dataset to obtain the indices of the selected features, and then using these selected feature indices, the near-optimal features are extracted from all samples of the second dataset to perform the classification task. For example, we have extracted the selected features obtained by employing HFS on the CeDF dataset (see Table 2) from the FF++ dataset using indices of selected features and performed the classification using those selected features only. The results are shown in Table 5. Following a similar approach, the classification performances on the CeDF dataset are recorded in Table 5. These results signify that the features selected using the proposed HFS are more effective in comparison to the actual features at least on these datasets used here.

4.5 Generalization capability of the proposed HFS method

We have conducted a set of experiments to test the generalization capability of the proposed HFS method. For this, we have conducted experiments in a cross-dataset setup used in [4, 6, 14]. The results are recorded in Table 6 along with some state-of-the-art works. In Table 6, the term CE_FF represents the experiment when the proposed

Table 4 p values of the Wilcoxon rank-sum test. Here Fet_1 to Fet_6 stands for Feature_1 to Feature_6. Also, Acc, AUC, and #Feature represent test accuracy, AUC score, and the number of features selected, respectively

	Dataset	Parameter	Fet_1	Fet_2	Fet_3	Fet_5	Fet_6
Fet_4	CeDF	Acc	0.000079	0.000079	0.000079	0.000079	0.000079
		AUC	0.000079	0.000334	0.034821	0.048152	0.017147
		#Feature	0.000079	0.000079	0.000079	0.000079	0.000079
	FF++	Acc	0.000079	0.001248	0.002036	0.001101	0.001600
		AUC	0.000079	0.000143	0.029309	0.001599	0.002035
		#Feature	0.000079	0.000079	0.000079	0.000079	0.000079

Table 5 Performances when HFS is employed on one dataset to obtain the near optimal feature subset and the indices of selected features present in the selected feature subset, and then only the selected features are extracted using these feature indices from another dataset for performing the classification task

Parameter	Feature_D	Feature_H	Feature_5	Feature_6	Feature_4
Case 1					
Test accuracy (in %)	95.37	65.64	96.72	96.72	97.30
AUC score (in %)	98.88	73.49	99.35	98.99	99.64
Number of features	2048	300	2348	804	297
Case 2					
Test accuracy (in %)	96.00	63.50	97.00	96.50	97.50
AUC score (in %)	98.34	70.22	97.83	98.68	99.22
Number of features	2048	300	2348	830	270

Here Case 1 represents the scenario when HFS employed on the FF++ to get indices of the features present in the selected feature subset and classification performed using only the selected features extracted from CeDF while Case 2 represents the opposite scenario

HFS-based deepfake detection model is trained on CeDF and tested on the FF++ dataset, while FF_CE represents the reverse case. It is worth mentioning here that the performances of the state-of-the-art methods are evaluated on our setups. From the results, we can safely conclude that the proposed method works well in the cross-dataset setup in comparison with state-of-the-art methods.

4.6 Comparison with state-of-the-art FS methods

In this work, we have designed a hybrid FS technique termed as GWO–VS method. All the experiments were conducted to compare the performance of the proposed hybrid FS technique with other state-of-the-art FS techniques. For the comparative purpose, we have considered five popularly used meta-heuristic methods: GA, PSO, HS, WOA, and GWO. For simplicity, we have concatenated

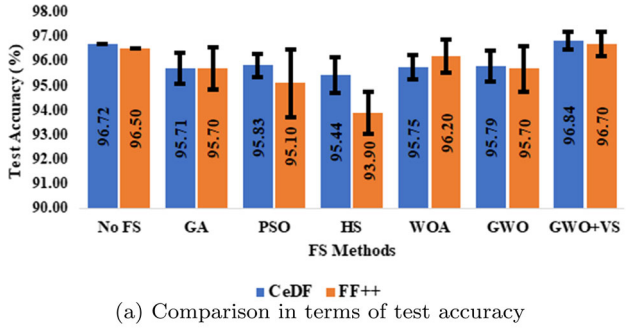
Table 6 Performance of the proposed model on a cross-dataset experimental setup

Exp.	Methods	Test accuracy (in %)	AUC score (in %)
	Li et al. [4]	64.50	75.19
	Afchar et al. [29]	50.50	51.51
	Qian et al. [56]	54.50	54.04
CE_FF	Mohiuddin et al. [6]	60.00	59.93
	Ganguly et al. [14]	65.00	63.80
	Proposed	66.50	76.72
	Li et al. [4]	58.06	55.60
	Afchar et al. [29]	66.41	65.58
	Qian et al. [56]	63.89	53.89
FF_CE	Mohiuddin et al. [6]	63.71	56.43
	Ganguly et al. [14]	68.04	66.12
	Zhao et al. [13]	–	67.44
	Miao et al. [12]	–	66.12
	Proposed	79.34	87.58

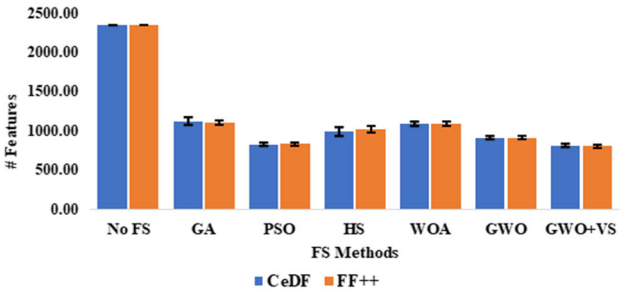
handcrafted features with deep learning features first and then employed all these FS techniques for comparison purposes. All these experiments are conducted 10 times and the results obtained are shown in Fig. 9. These comparative results show that, in terms of average test accuracy and the number of selected features, the proposed hybrid FS outperforms other FS methods used here for comparison. Moreover, the better performance of the GWO–VS-based FS technique over the GWO-based FS justifies the choice of GWO–VS over the GWO algorithm within the proposed HFS methods.

4.6.1 Statistical test among different FS proposals

We have also performed a statistical test to check whether the result of the HFS method is statistically significant with respect to the other FS methods used here for comparison. The objective of the test is to check whether there is enough evidence to "reject" the null hypothesis: there is no statistically significant difference between the performances of HFS and another method. Here we have used the non-parametric statistical test known as Wilcoxon rank-sum test following the approach described in [55]. According to this test, if the estimated p value <0.05, then we reject the null hypothesis at a 5% significance level. To calculate the p value, we performed 10 independent trials for all these FS methods and the estimated p values are recorded in Table 7. From the estimated p values we can infer that the null hypothesis can be rejected i.e., the higher performance obtained by the proposed HFS method is statistically significant.



(a) Comparison in terms of test accuracy



(b) Comparison in terms of feature count

Fig. 9 Comparison of accuracies for various state-of-the-art FS methods on CeDF and FF++ datasets. Here, No FS indicates that classification is performed using the original features (i.e., Feature_5)

4.7 Comparison with state-of-the-art deepfake detection methods

We have compared the performances of the proposed deepfake detection method with the following state-of-the-art deepfake methods [4, 5, 12, 13, 29, 56]. To avoid any bias imposed by different evaluation processes, all the methods except [12, 13] have been evaluated on the current dataset splits (see Table 1) and using our own experimental setup. Table 8 lists the comparative performances of all the methods. Table 10 shows the image level detection results of these state-of-the-art models. In Table 10, the terms 'Yes' and 'No' indicate whether the image under consideration is correctly and incorrectly recognized by the underlying method, respectively, whereas the symbol '–' indicates that the corresponding method has not been evaluated on this dataset. The results show that the performance of the proposed method is comparable with the state-of-the-art methods. The proposed method outperforms all the methods used here for comparison on the CeDF dataset but fails to achieve state-of-the-art results in the FF++ dataset in terms of the AUC score. We would also like to mention that our experimental setups are different from the works proposed by Zhao et al. [13] and Miao et al. [12] in terms of the number of samples used for training and testing, preparing cropped face images, the number of frames per video, etc. which might be the reason for worse performance as compared to these works.

4.8 Performance on the DFDC dataset

The Deepfake Detection Challenge (DFDC) dataset was introduced in a Kaggle contest [54] and later made public by Facebook AI [57], which is one of the most comprehensive third-generation forensics datasets to date. The training, validation, public test, and private test sets consist of 119,154, 4000, 5000, and 10000 clips (10 s each), respectively. In our work, we have used 86,166 video clips out of 119,154 available for training our model due to computational constraints. This subset is obtained by

Table 7 Estimated p values using Wilcoxon rank-sum test

Dataset	GA	PSO	HS	WOA	GWO
CeDF	0.0020	0.0031	0.0014	0.0002	0.0001
FF++	0.0005	0.0015	0.0005	0.0001	0.0001

Table 8 Performance of the proposed model on the intra-dataset experimental setup

Dataset	Methods	Model	# Features	Test accuracy (in %)	AUC score (in %)
	Li et al. [4]	XceptionNet	2048	95.37	98.88
	Afchar et al. [29]	MesoNet	1024	65.64	65.33
	Afchar et al. [29]	MesoInception	1024	65.83	64.80
CeDF	Qian et al. [56]	F3-Net	4096	87.06	81.48
	Guo et al. [5]	AMTEN	300	68.33	78.04
	Proposed	HFS	270	**97.30**	**99.35**
	Li et al. [4]	XceptionNet	2048	96.00	98.34
	Afchar et al. [29]	MesoNet	1024	67.50	66.81
	Afchar et al. [29]	MesoInception	1024	65.00	66.93
FF++	Qian et al. [56]	F3-Net	4096	95.50	96.52
	Guo et al. [5]	AMTEN	300	78.50	87.62
	Zhao et al. [13]	Multi-Attention + XceptionNet	–	96.37	98.97
	Zhao et al. [13]	Multi-Attention + EfficientNet-B4	–	97.60	99.29
	Miao et al. [12]	FRLM	–	97.63	**99.50**
	Proposed	HFS	297	**98.00**	99.16

Bold values describe the maximum accuracy and AUC reached by the corresponding method

Table 9 Performance of the proposed HFS model and its components (see Fig. 6) on DFDC dataset

Parameter	Feature_H	Feature_D	Feature_1	Feature_2	Feature_3	Feature_4	Feature_5	Feature_6
Test accuracy (in %)	49.91	69.19	51.91	72.03	66.83	75.34	71.81	73.07
AUC score (in %)	51.01	77.59	53.66	79.88	75.63	85.67	80.19	81.24
# Features	300	2048	102	653	755	240	2348	859

considering only the first frame of each video clip in the training set. However, we have used the whole testing set of 5000 public video clips to evaluate the performance of our model. We have also used all the videos from the validation set to evaluate objective function values during the FS phase. The experimental results are recorded in Table 9. We have also reported the performances of four state-of-the-art methods [5, 6, 9, 14] (publicly available codes were used) in Table 11. Table 10 shows the image-level detection results of the state-of-the-art models on the DFDC dataset. From these results, it can be observed that the performance of the proposed method is comparable with state-of-the-art methods considered here for comparison (Table 11).

5 Conclusion

In recent times, the detection of deepfake videos and images has become a challenging task for researchers because of their striking resemblance to the real ones. In this paper, an HFS-based method is proposed, which considers both handcrafted and deep learning features obtained from the input, and selects only important feature elements by ignoring the irrelevant ones. Moreover, the proposed HFS method allows for a drastic reduction of the dimensions of the descriptors with a high impact on the performance of the system. An exhaustive set of experiments has been conducted on three benchmark datasets, namely CeDF, FF++, and DFDC, and obtained 99.35%, 99.16%, and 85.67% AUC scores, respectively. The results achieved by the proposed method are comparable with most state-of-the-art methods considered here for comparison. However, there are still some rooms to improve the proposed method as its performances are not much satisfactory, especially when considering the cross-dataset

Table 10 Comparison with state-of-the-art methods at image level

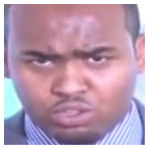
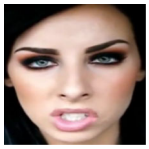
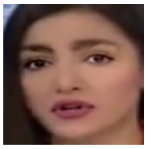
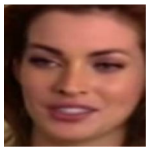
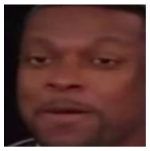
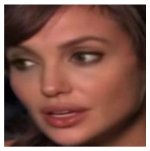
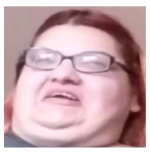
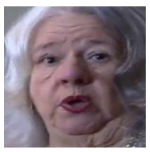
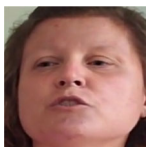
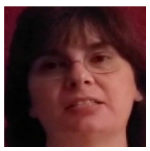
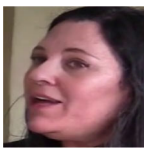
Method	Real			Fake		
						
FF++						
Li et al. [4]	Yes	Yes	No	Yes	Yes	No
Afchar et al. [29]	Yes	Yes	No	Yes	No	No
Guo et al. [5]	Yes	No	Yes	No	Yes	No
Mohiuddin et al. [6]	Yes	Yes	No	No	Yes	No
Ganguly et al. [14]	No	Yes	No	Yes	Yes	No
Proposed	Yes	Yes	No	Yes	Yes	No
Method	Real			Fake		
						
CeDF						
Li et al. [4]	No	Yes	No	Yes	No	No
Afchar et al. [29]	No	Yes	No	No	Yes	No
Guo et al. [5]	Yes	No	Yes	Yes	No	No
Mohiuddin et al. [6]	Yes	No	Yes	No	No	Yes
Ganguly et al. [14]	Yes	No	Yes	No	Yes	No
Proposed	Yes	Yes	Yes	Yes	No	Yes
Method	Real			Fake		
						
DFDC						
Li et al. [4]	–	–	–	–	–	–
Afchar et al. [29]	–	–	–	–	–	–
Guo et al. [5]	No	No	Yes	Yes	No	Yes
Mohiuddin et al. [6]	No	No	Yes	Yes	Yes	No
Ganguly et al. [14]	No	Yes	No	Yes	Yes	Yes
Proposed	Yes	Yes	Yes	Yes	No	Yes

Table 11 Performance comparison of different models tested on the DFDC dataset

Method	Performance (in %)	
	AUC score	Test accuracy
Guo et al. [5]	74.19	**76.82**
Mohiuddin et al. [6]	71.77	71.30
Ganguly et al. [14]	83.59	75.20
Ganguly et al. [9]	**86.32**	73.21
Proposed	85.67	75.34

Bold values describe the maximum accuracy and AUC reached by the corresponding method

scenario, by considering unpleasant effects on society caused by deepfake videos/images. The base model XceptionNet can be replaced by other CNN models to check the performance. In the future, attention-based CNN networks can also be tried as deep learning feature extractors. Besides, the proposed method can be applied to other datasets to test its robustness.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/s00521-023-08201-z.

Acknowledgements We are thankful to the Center for Microprocessor Applications for Training Education and Research (CMATER) research laboratory of the Computer Science and Engineering Department, Jadavpur University, Kolkata, India for providing infrastructural support.

Funding The authors gratefully acknowledge financial support from ANID Fondecyt 1231122, PIA/PUENTE AFB220003.

Data availability statement All the data used in this work are publicly available to the researchers through the works [4, 28, 54].

Declarations

Conflict of interest All the authors declare that they have no conflict of interest or competing interest.

References

1. McCloskey S, Albright M (2019) Detecting GAN-generated imagery using saturation cues. In: IEEE international conference on image processing (ICIP). IEEE, pp 4584–4588
2. Durall R, Keuper M, Pfreundt FJ, Keuper J (2019) Unmasking deepFakes with simple features. arXiv preprint arXiv:1911.00686
3. Li Y, Chang MC, Lyu S (2018) In ictu oculi: exposing AI generated fake face videos by detecting eye blinking. In: IEEE international workshop on information forensics and security (WIFS). IEEE, pp 1–7
4. Li Y, Yang X, Sun P, Qi H, Lyu S (2020) Celeb-DF: a large-scale challenging dataset for deepfake forensics. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 3207–3216
5. Guo Z, Yang G, Chen J, Sun X (2021) Fake face detection via adaptive manipulation traces extraction network. Comput Vis Image Underst 204:103170
6. Mohiuddin S, Ganguly S, Malakar S, Kaplun D, Sarkar R (2022) A feature fusion based deep learning model for deepfake video detection. In: International conference on mathematics and its applications in new computer systems. Springer, pp 197–206
7. Chollet F (2017) Deep learning with depthwise separable convolutions. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 1251–1258
8. Selvaraju RR, Cogswell M, Das A, Vedantam R, Parikh D, Batra D (2017) Grad-CAM: visual explanations from deep networks via gradient-based localization. In: Proceedings of the IEEE international conference on computer vision, pp 618–626
9. Ganguly S, Ganguly A, Mohiuddin S, Malakar S, Sarkar R (2022) ViXNet: vision transformer with Xception Network for deepfakes based video and image forgery detection. Expert Syst Appl 210:118423
10. Wang C, Deng W (2021) Representative forgery mining for fake face detection. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 14923–14932
11. Chen S, Yao T, Chen Y, Ding S, Li J, Ji R (2021) Local relation learning for face forgery detection. In: Proceedings of the AAAI conference on artificial intelligence, pp 1081–1088
12. Miao C, Chu Q, Li W, Li S, Tan Z, Zhuang W et al (2021) Learning forgery region-aware and ID-independent features for face manipulation detection. IEEE Trans Biom Behav Ident Sci 4(1):71–84
13. Zhao H, Zhou W, Chen D, Wei T, Zhang W, Yu N (2021) Multi-attentional deepfake detection. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 2185–2194
14. Ganguly S, Mohiuddin S, Malakar S, Cuevas E, Sarkar R (2022) Visual attention based deepfake video forgery detection. Pattern Anal Appl 25:1–12
15. Das S, Chatterjee A, Dey S, Saha S, Malakar S (2023) Breast cancer detection from histology images using deep feature selection. In: Basu S, Kole DK, Maji AK, Plewczynski D, Bhattacharjee D (eds) Proceedings of international conference on frontiers in computing and systems: COMSYS 2021. Springer, Berlin, pp 323–330
16. Banerjee D, Chatterjee B, Bhowal P, Bhattacharyya T, Malakar S, Sarkar R (2021) A new wrapper feature selection method for language-invariant offline signature verification. Expert Syst Appl 186:115756
17. Ghosh M, Malakar S, Bhowmik S, Sarkar R, Nasipuri M (2017) Memetic algorithm based feature selection for handwritten city name recognition. In: International conference on computational intelligence, communications, and business analytics. Springer, pp 599–613
18. Bommert A, Sun X, Bischl B, Rahnenführer J, Lang M (2020) Benchmark for filter methods for feature selection in high-dimensional classification data. Comput Stat Data Anal 143:106839
19. Tola E, Lepetit V, Fua P (2009) DAISY: an efficient dense descriptor applied to wide-baseline stereo. IEEE Trans Pattern Anal Mach Intell 32(5):815–830
20. Chatterjee A, Malakar S, Sarkar R, Nasipuri M (2018) Handwritten digit recognition using DAISY descriptor: a study. In: 2018 Fifth international conference on emerging applications of information technology (EAIT). IEEE, pp 1–4
21. Majumder S, Ghosh S, Malakar S, Sarkar R, Nasipuri M (2021) A voting-based technique for word spotting in handwritten document images. Multimed Tools Appl 80(8):12411–12434
22. Malakar S, Ghosh M, Bhowmik S, Sarkar R, Nasipuri M (2019) A GA based hierarchical feature selection approach for

handwritten word recognition. Neural Comput Appl 32(7):2533–2552. https://doi.org/10.1007/s00521-018-3937-8
23. Mirjalili S, Mirjalili SM, Lewis A (2014) Grey wolf optimizer. Adv Eng Softw 69:46–61
24. Doğan B, Ölmez T (2015) A new metaheuristic for numerical function optimization: vortex Search algorithm. Inf Sci 293:125–145. https://doi.org/10.1016/j.ins.2014.08.053
25. Matern F, Riess C, Stamminger M (2019) Exploiting artifacts visual, to expose deepfakes and face manipulations. In: IEEE winter applications of computer vision workshops (WACVW). IEEE, pp 83–92
26. Yang X, Li Y, Lyu S (2019) Exposing deep fakes using inconsistent head poses. In: ICASSP 2019–2019 IEEE international conference on acoustics, speech and signal processing (ICASSP). IEEE, pp 8261–8265
27. Tolosana R, Romero-Tapiador S, Fierrez J, Vera-Rodriguez R (2021) DeepFakes evolution: analysis of facial regions and fake detection performance. In: International conference on pattern recognition. Springer, pp 442–456
28. Rossler A, Cozzolino D, Verdoliva L, Riess C, Thies J, Nießner M (2019) Faceforensics++: learning to detect manipulated facial images. In: Proceedings of the IEEE/CVF international conference on computer vision, pp 1–11
29. Afchar D, Nozick V, Yamagishi J, Echizen I (2018) MesoNet: a compact facial video forgery detection network. In: IEEE international workshop on information forensics and security (WIFS). IEEE, pp 1–7
30. Amerini I, Galteri L, Caldelli R, Del Bimbo A (2019) Deepfake video detection through optical flow based CNN. In: Proceedings of the IEEE/CVF international conference on computer vision workshops
31. Dang H, Liu F, Stehouwer J, Liu X, Jain AK (2020) On the detection of digital face manipulation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 5781–5790
32. Tolosana R, Vera-Rodriguez R, Fierrez J, Morales A, Ortega-Garcia J (2020) Deepfakes and beyond: a survey of face manipulation and fake detection. Inf Fusion 64:131–148
33. Mirsky Y, Lee W (2021) The creation and detection of deepfakes: a survey. ACM Comput Surv (CSUR) 54(1):1–41
34. Shaw SS, Ahmed S, Malakar S, Garcia-Hernandez L, Abraham A, Sarkar R (2021) Hybridization of ring theory-based evolutionary algorithm and particle swarm optimization to solve class imbalance problem. Complex Intell Syst 7(4):2069–2091
35. Malakar S, Ghosh M, Chaterjee A, Bhowmik S, Sarkar R (2020) Offline music symbol recognition using Daisy feature and quantum Grey wolf optimization based feature selection. Multimed Tools Appl 79(43):32011–32036
36. Sarkar S, Ghosh M, Chatterjee A, Malakar S, Sarkar R (2018) An advanced particle swarm optimization based feature selection method for tri-script handwritten digit recognition. In: International conference on computational intelligence, communications, and business analytics. Springer, pp 82–94
37. Davis L (1991) Handbook of genetic algorithms. In: CumInCAD
38. Simon D (2008) Biogeography-based optimization. IEEE Trans Evol Comput 12(6):702–713
39. Qin AK, Huang VL, Suganthan PN (2008) Differential evolution algorithm with strategy adaptation for global numerical optimization. IEEE Trans Evol Comput 13(2):398–417
40. Kennedy J, Eberhart R (1995) Particle swarm optimization. In: Proceedings of ICNN'95-international conference on neural networks, vol 4. IEEE, pp 1942–1948
41. Yang XS, Deb S (2009) Cuckoo search via Lévy flights. In: World congress on nature and biologically inspired computing (NaBIC). IEEE, pp 210–214
42. Karaboga D (2010) Artificial bee colony algorithm. Scholarpedia 5(3):6915
43. Goffe WL, Ferrier GD, Rogers J (1994) Global optimization of statistical functions with simulated annealing. J Econom 60(1–2):65–99
44. Geem ZW, Kim JH, Loganathan GV (2001) A new heuristic optimization algorithm: harmony search. Simulation 76(2):60–68
45. Rashedi E, Nezamabadi-pour H, Saryazdi S (2009) GSA: a gravitational search algorithm. Inf Sci 179(13):2232–2248
46. Ahmed S, Ghosh KK, Garcia-Hernandez L, Abraham A, Sarkar R (2020) Improved coral reefs optimization with adaptive β-hill climbing for feature selection. Neural Comput Appl 33(12):6467–6486. https://doi.org/10.1007/s00521-020-05409-1
47. Glover F, Laguna M (1998) Tabu search. In: Du D-Z, Pardalos PM (eds) Handbook of combinatorial optimization. Springer, Berlin, pp 2093–2229
48. Alomari OA, Khader AT, Al-Betar MA, Awadallah MA (2018) A novel gene selection method using modified MRMR and hybrid bat-inspired algorithm with β-hill climbing. Appl Intell 48(11):4429–4447. https://doi.org/10.1007/s10489-018-1207-1
49. Wang Z, Wu G, Wan Z (2017) A novel hybrid vortex search and artificial bee colony algorithm for numerical optimization problems. Wuhan Univ J Nat Sci 22(4):295–306
50. Chatterjee B, Bhattacharyya T, Ghosh KK, Singh PK, Geem ZW, Sarkar R (2020) Late acceptance hill climbing based social ski driver algorithm for feature selection. IEEE Acces 8:75393–75408. https://doi.org/10.1109/access.2020.2988157
51. Dey C, Bose R, Ghosh KK, Malakar S, Sarkar R (2022) LAGOA: learning automata based grasshopper optimization algorithm for feature selection in disease datasets. J Ambient Intell Humaniz Comput 13(6):3175–3194
52. Jiang B, Ren Q, Dai F, Xiong J, Yang J, Gui G (2018) Multi-task cascaded convolutional neural networks for real-time dynamic face recognition method. In: International conference in communications, signal processing, and systems. Springer, pp 59–66
53. Alzer H (1997) On some inequalities for the incomplete gamma function. Math Comput 66(218):771–778
54. Dolhansky B, Howes R, Pflaum B, Baram N, Ferrer CC (2019) The deepfake detection challenge (DFDC) preview dataset. arXiv preprint arXiv:1910.08854
55. Mondal R, Malakar S, Barney Smith EH, Sarkar R (2022) Handwritten English word recognition using a deep learning based object detection architecture. Multimed Tools Appl 81(1):975–1000
56. Qian Y, Yin G, Sheng L, Chen Z, Shao J (2020) Thinking in frequency: face forgery detection by mining frequency-aware clues. In: European conference on computer vision. Springer, pp 86–103
57. Dolhansky B, Bitton J, Pflaum B, Lu J, Howes R, Wang M et al (2020) The deepfake detection challenge (DFDC) dataset. arXiv preprint arXiv:2006.07397