FOREX Publication
Open Access | Rapid and quality publishing

International Journal of
Electrical and Electronics Research (IJEER)
Research Article | Volume 12, Issue 2 | Pages 696-710 | e-ISSN: 2347-470X

Deepfake Detection using Integrate-Backward-Integrate Logic Optimization Algorithm with CNN

R. Uma Maheshwari[1*], B. Paulchamy[2], Arun M[3], Vairaprakash Selvaraj[4], Dr. N. Naga Saranya[5] and Dr. Sankar Ganesh S[6]

[1]*Hindusthan Institute of Technology, Coimbatore, India; umamaheshwari@hit.edu.in*
[2]*Professor, Hindusthan Institute of Technology, Coimbatore, India; luckshanthpaul@gmail.com*
[3]*Assistant Professor, Department of ECE, Panimalar Engineering College, Chennai, India; arunmemba@ieee.org*
[4]*Associate Professor, Department of Electronics and Communication Engineering Ramco Institute of Technology, India; vairaprakashklu@gmail.com*
[5]*Associate Professor, Department of Computer Applications, Saveetha College of Liberal Arts & Sciences, SIMATS, Chennai India; drnagasaranya@gmail.com*
[6]*Associate professor, Department of Electronics and Communication Engineering, AVS Engineering college, Millitary Road, Ammapet, Salem, Tamilnadu, India; sankar.ganesh308@gmail.com*

***Correspondence:** R. Uma Maheshwari: *umamaheshwari@hit.edu.in;*

ABSTRACT- The emergence of deepfake technology has spurred the need for robust and adaptive methods to detect manipulated media content. This study explores the integration of the Integrate-backward-integrate (IbI) Logic Optimization Algorithm with Convolutional Neural Networks (CNNs) for enhanced deepfake detection. The proposed approach involves a multi-phase iterative process: the CNN initially trained on a diverse dataset encompassing both real and deepfake images. The CNN serves as the foundation for the IbI-driven optimization. The integration phase employs the trained CNN to forward-integrate images, classifying them as real or deepfake. Subsequently, the IbI Logic Optimization Algorithm engages in the backward phase, utilizing feedback from the CNN's performance to iteratively refine the network's parameters, architecture, and feature extraction capabilities. This iterative optimization process aims to adaptively enhance the CNN's ability to discern subtle nuances between authentic and manipulated visuals. The re-integration phase evaluates the refined CNN's performance through multiple iterations, seeking to iteratively improve deepfake detection accuracy. Validation occurs using separate datasets to prevent overfitting and ensure the model's generalizability. The proposed method aims to enhance the CNN's adaptability to evolving deepfake techniques, addressing the dynamic nature of manipulative media creation. This fusion of IbI Logic Optimization with CNNs presents a promising avenue for bolstering deepfake detection capabilities. However, the effectiveness of this approach relies on dataset quality, network architecture, and the dynamic nature of deepfake generation techniques. Continuous refinement and validation are essential to adapt the model to new challenges posed by advancing deepfake technologies.

Keywords: Deepfake Detection, Integrate-backward-integrate (IbI), Convolutional Neural Networks (CNNs), Image Manipulation, Iterative Optimization, Adaptive Learning.

ARTICLE INFORMATION

Author(s): R. Uma Maheshwari, B. Paulchamy, Arun M, Vairaprakash Selvaraj, Dr. N. Naga Saranya and Dr. Sankar Ganesh S;

Received: 27/02/2024; **Accepted**: 28/05/2024; **Published**: 28/06/2024;
e-ISSN: 2347-470X;
Paper Id: IJEER 2702-24;
Citation: 10.37391/IJEER.120248
Webpage-link:
https://ijeer.forexjournal.co.in/archive/volume-12/ijeer-120248.html

Publisher's Note: FOREX Publication stays neutral with regard to Jurisdictional claims in Published maps and institutional affiliations.

1. INTRODUCTION

Digital videos are what you'd see on modern televisions, movie theatre screens, cell phones, computers, *etc.* Many codec approaches are used to compress the digital films, including MPEG-2, MPEG-4, H.264[1], H.265, [2], and H.266 [3]. Because they are saved on disks and Blu-ray discs, these digital films are quite portable. Even low-end devices can readily record most digital videos. One of the main reasons why low-end gadget videos are so prevalent on social media is because of how convenient they are. Nevertheless, Analog technology was used to record the classic videos that were seen on cathode ray tube systems TVs. A larger and heavier video cassette recorder was required to view or play back these analogue movies recorded on magnetic tapes. However, when contrasted with digital videos, analog ones are more reliable. Any kind of modification in an analog video demands powerful and efficient gear. Not only are these analog alterations challenging to execute, but they are also readily discernible to the naked eye with close inspection. But because digital movies are so easy to modify, anybody with access to the right software and technology can fake them. Untrustworthy in comparison to analog videos.

Due to the proliferation of low-cost technology, the relative simplicity of video creation, and the abundance of free editing tools, digital video editing software continues to see meteoric

FOREX Publication
Open Access | Rapid and quality publishing
International Journal of Electrical and Electronics Research (IJEER)
Research Article | Volume 12, Issue 2 | Pages 696-710 | e-ISSN: 2347-470X

growth and has become very popular even among non-professionals. The widespread availability of training videos on major platforms like YouTube etc. further contributes to the ease with which regular people may run video-editing tools. The latest smartphone models come with basic video editing tools including filters that let you apply effects, improve contrast and clarity, merge footage, and more. Nevertheless, with the help of advanced video editing software, you may do tasks like adding motion effects, changing color graphics, cutting video clips, combining 1 video clip, etc. [4] In addition, expert video editing software may help create a smooth transition between the two locations. They seem to be in the same spot because the digital transition video snippets are that crisp. Video editing software and tools have recently begun to include deep learning networks.

Deepfakes, a combination of the words "deep learning" and "fake," pose a serious threat in this age of artificially generated content. These artificial works of art, often directed by complex Generative Adversarial Networks (GANs), use deep learning methods to create material that looks and acts much like the real thing. Facial manipulations, lip-syncing, speech synthesis, and even subtle behavioral reactions are all part of this domain of digital deceit. The difficulty of identifying deepfakes is rising in tandem with the speed at which their underlying technology is developing. To effectively address these difficulties, it is crucial to have a thorough grasp of the terrain.

The fast development of AI, particularly GANs, has been a major driving force behind the creation of deepfakes [5], since it allows for the fabrication of synthetic material that is very lifelike. The authenticity and diversity of these modifications, which may outstrip the discerning capabilities of conventional detection systems, is one of the main obstacles. With the help of pre-trained models, transfer learning makes things even more complicated by allowing the manipulation of pieces to be seamlessly integrated into created content. Due to the prevalence of techniques such as lip-syncing, audio manipulation, and face swapping, identification based just on facial traits is inadequate. There has to be constant innovation in detection systems because to the ongoing arms race between deepfake developers and detection mechanisms. To combat this, academics are using multimodal techniques, behavioural analysis, and deep learning models to decipher the complex characteristics of deepfake material. This deepfake mystery is complex on many levels, and the difficulty level is high since it goes beyond individual frames and requires a comprehensive understanding of time and space. In this day where the boundaries between fact and fiction are becoming porous, it is critical to address these intricacies in order to protect the credibility of media material.

The emergence of deepfake technology presents a serious danger to the validity of digital material, exacerbating problems linked to disinformation, privacy violation, and the loss of confidence in multimedia. Because deepfake developers use more complex approaches, established methods of detection are frequently unable to keep up [6]. The need for sophisticated, reliable solutions that can successfully detect deepfake manipulations is what inspired the suggested method, which use the Integrate-backward-integrate Logic Optimization Algorithm in combination with Convolutional Neural Networks (CNNs). To tackle the many nuances involved in both deepfake content production and detection, we want to improve detection systems' accuracy and reliability by combining deep learning with a logic optimization approach.

In a groundbreaking move, the suggested Deepfake Detection framework combines CNNs with the Integrate-backward-integrate Logic Optimization Algorithm. The special contribution is the combination of convolutional neural networks (CNNs), known for their capacity to recognize intricate visual patterns, and logic optimization, which is great at picking up on little anomalies in altered information. The goal of this combined strategy is to improve detection accuracy by making use of the best features of both approaches.

By adding the Integrate-backward-integrate Logic Optimization Algorithm, deepfake detection gains a new level of granularity, allowing for the discovery of irregularities and complex patterns that could otherwise go undetected. With the CNN component, the model can learn hierarchical features better, which means it can analyze visual input more nuancedly, especially in video frames.

The significance of this methodology goes beyond traditional detection methods since it tackles the problems caused by deepfake technology via this novel combination. The objective is to develop stronger definitions that can withstand ever-changing deepfake strategies, promoting greater confidence and dependability in this age of digital media when identifying genuine content is crucial. Our digital world is always being threatened by deceitful synthetic media, but our comprehensive strategy aims to establish a new standard in this fight.

The organization of paper is as follows; *section 2* includes literature survey of existing work; *section 3* includes methodology of proposed work; *section 4* includes experimental results and analysis; *section 5* includes conclusion and future work.

2. LITERATURE SURVEY

An examination of the literature in the ever-evolving area of deepfake detection [7] demonstrates substantial progress in comprehending and overcoming the obstacles presented by synthetic media. Recent research, such as the review on logic-based techniques by Gupta et al., highlights the possibility of using reasoning to improve the precision of detection systems. In order to understand the suggested integration with the Integrate-backward-integrate Logic Optimization Algorithm, it is necessary to first understand the function of convolutional neural networks (CNNs), which Anwar et al. discuss in their work on deepfake detection.

In addition, the roadmap for the suggested approach's relevance and prospective contributions is provided by the study [7] that discusses current issues and future approaches in deepfake detection. Zhou et al.'s investigation of detection strategies encapsulates the ever-changing nature of deepfake technology,

FOREX Publication
Open Access | Rapid and quality publishing
International Journal of Electrical and Electronics Research (IJEER)
Research Article | Volume 12, Issue 2 | Pages 696-710 | e-ISSN: 2347-470X

highlighting the ongoing need for creative solutions to keep up with deceitful innovations.

By combining convolutional neural networks (CNNs) with the Integrate-backward-integrate Logic Optimization Algorithm, we hope to tackle the deepfake problem's complexities, which the literature has shown to be complex and multi-faceted. This technique seeks to provide a fresh viewpoint to the continuing discussion on successful deepfake detection by combining deep learning [8] capabilities with logical reasoning. The literature review provides a solid groundwork, but the suggested integration takes things a step further towards finding solutions that can withstand and adapt to the ongoing threats from deepfake technology.

The studied literature on deepfake detection sheds light on different approaches and obstacles. Nevertheless, there is a noticeable lack of study on deepfake detection using the Integrate-backward-integrate Logic Optimization Algorithm in conjunction with Convolutional Neural Networks (CNNs). Research specifically addressing the synergistic potential of logic optimization and deep learning for this particular application is scarce, despite the fact that the assessed publications provide thorough overviews of logic-based methods, CNNs, and the larger landscape of detection techniques.

Table 1. Comparison with Existing Methodology

Methodology	Key Features	Strengths	Limitations
CNN-based Approaches [9]	- Utilizes deep learning for feature extraction	- Effective in capturing spatial dependencies	- Limited temporal analysis
RNN-based Approaches [10]	- Captures temporal dependencies in video sequences	- Effective for analysing dynamic patterns	- May struggle with long-range dependencies
GAN-aware Detection [11]	- Exploits artifacts introduced by GANs	- Robust against sophisticated deepfakes	- High false positive rates for some authentic content
Audio-Visual Fusion [12]	- Integrates analysis of both visual and audio cues	- More robust against multimodal manipulations	- Requires synchronization of audio and video data
Behavioural Analysis [13]	- Analyses subtle behavioural cues in videos	- Effective against contextually aware fakes	- Limited by the availability of behavioural datasets
Feature-based Methods [14]	- Extracts handcrafted features for detection	- May be interpretable and computationally light	- Less adaptive to evolving deepfake creation methods
Logic-based Approaches [15]	- Applies logical reasoning for anomaly detection	- Captures inconsistencies in synthesized content	- May struggle with complex, context-aware deepfakes
Ensemble Learning [16]	- Combines predictions from multiple models	- Improves overall detection robustness	- Increased computational complexity
Blockchain-based Solutions [17]	- Utilizes blockchain for media authentication	- Provides traceability transparency	- Requires widespread adoption for effectiveness

The proposed integration signifies a novel and innovative approach that combines logical reasoning with the powerful feature extraction capabilities of CNNs. The research gap lies in the absence of in-depth investigations into the effectiveness, limitations, and unique contributions of this specific amalgamation in the context of deepfake detection. Researchers and practitioners interested in enhancing the robustness of deepfake detection systems may find limited guidance in the existing literature regarding the intricacies and potential challenges associated with incorporating the Integrate-backward-integrate Logic Optimization Algorithm into a CNN-based framework.

To address this research gap, future studies could delve into empirical evaluations, comparative analyses, and case studies that specifically assess the performance of the proposed integration. Additionally, investigations into how this hybrid approach [18] handles variations in deepfake creation techniques, such as face swapping, lip-syncing, and audio manipulations, would contribute to a more comprehensive understanding of its applicability. Bridging this gap would not only advance the theoretical understanding of deepfake detection but also provide practical insights for the development of more effective and adaptive detection systems in the face of evolving synthetic media technologies.

The significance of developing and refining methodologies for deepfake detection, especially those integrating novel approaches like the Integrate-backward-integrate Logic Optimization Algorithm with Convolutional Neural Networks (CNNs), lies in addressing critical challenges associated with synthetic media.

Deepfakes pose a significant threat to the trustworthiness of digital content. The development of effective detection methodologies is crucial for preserving the authenticity of media, mitigating the potential damage caused by deceptive manipulations.

Deepfakes have the potential to be exploited for spreading misinformation and disinformation. Robust detection methods

FOREX Publication
Open Access | Rapid and quality publishing
International Journal of Electrical and Electronics Research (IJEER)
Research Article | Volume 12, Issue 2 | Pages 696-710 | e-ISSN: 2347-470X

contribute to mitigating the impact of manipulated content on public perception and discourse.
Deepfake technology can be used for malicious purposes, including identity theft and impersonation. Detection methods play a pivotal role in safeguarding individuals, public figures, and organizations from reputational and financial harm. Deepfakes can be used to create fabricated content that invades personal privacy. Detection techniques help in identifying and mitigating privacy infringements, ensuring individuals have control over their digital representations.

As deepfakes [19] become more sophisticated, the need for robust cybersecurity measures intensifies. Advanced detection methodologies contribute to bolstering cybersecurity by identifying and thwarting potential threats rooted in synthetic media. The proposed integration of the Integrate-backward-integrate Logic Optimization Algorithm with CNNs signifies adaptability to evolving deepfake creation methods. This adaptability is crucial for staying ahead in the cat-and-mouse game between deepfake creators and detection mechanisms.

Research and development in deepfake detection drive technological innovation. The exploration of novel methodologies, such as the proposed integration, contributes to advancing the field and developing more sophisticated and effective detection tools. Governments and regulatory bodies are increasingly recognizing the threats posed by deepfakes. Robust detection methodologies play a role in ensuring compliance with regulations and holding malicious actors accountable for their actions.

The ability to detect and mitigate deepfake threats contributes to building resilience in digital environments. This resilience is essential for maintaining the integrity of digital communications, entertainment, and information dissemination. The development and deployment of AI-based solutions, such as deepfake detection [20] methods, underscore the importance of responsible and ethical use of artificial intelligence. This contributes to fostering a positive and secure digital ecosystem.

3. PROPOSED METHODOLOGY

The proposed methodology for deepfake detection integrates the Integrate-backward-integrate (IbI) Logic Optimization Algorithm with Convolutional Neural Networks (CNNs) to create a dynamic and adaptive approach to counter the challenges posed by deepfake technology. The methodology unfolds in multiple phases to iteratively enhance the CNN's ability to distinguish between authentic and manipulated visuals.

Before feeding images into a deep neural network, preprocessing steps are often applied, such as resizing, normalization, and data augmentation. These steps help ensure that the input data is in a suitable format for the network.

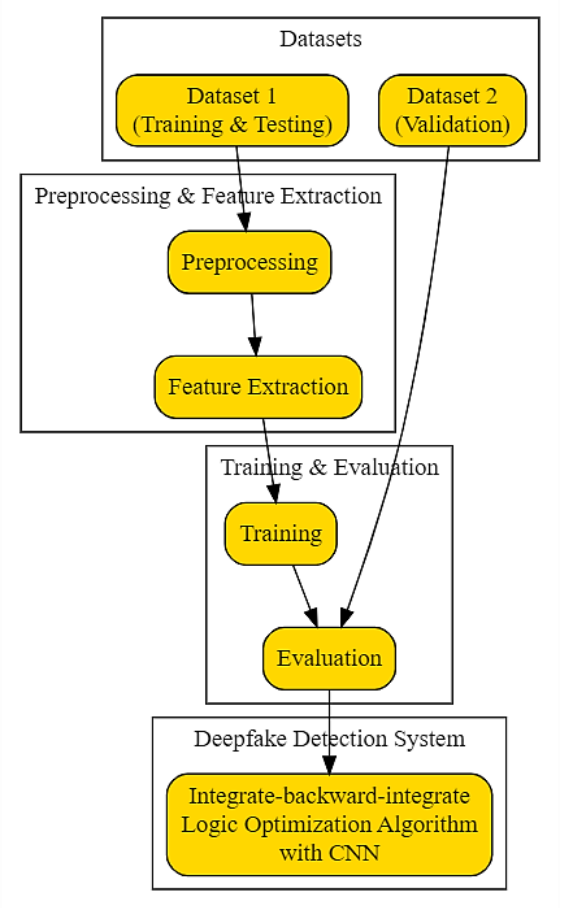

Figure 1. Deepfake Detection process

3.1. Training the CNN

In the phase of Forward Integration with the Convolutional Neural Network (CNN) for deepfake detection, the trained model processes input images through a series of mathematical operations to make predictions about their authenticity. The initial phase involves training the CNN on a diverse dataset that includes both real and deepfake images. This foundational training equips the CNN with the capability to recognize patterns and features indicative of manipulated content.

Let X represent the input image, W denote the learned weights of the CNN's filters, and b signify the bias terms. The forward pass involves a series of operations, starting with convolution, activation, pooling, and eventually leading to the classification of the input image. The convolutional layers, characterized by the equation Convolutional Neural Networks (CNNs) are commonly used for face recognition tasks. The convolutional layers apply filters to the input images, extracting hierarchical features. The output of these layers can be calculated using the convolution operation:

$$Z(i,j) = \sum_m \sum_n X(i+m, j+n) \cdot W(m,n) + b \quad (1)$$

where Z(i,j) is the output, X(i+m,j+n) is the input pixel value, W(m,n) is the filter weight, and b is the bias term convolve the input image with learned filters to extract hierarchical features.

FOREX Publication
Open Access | Rapid and quality publishing
International Journal of Electrical and Electronics Research (IJEER)
Research Article | Volume 12, Issue 2 | Pages 696-710 | e-ISSN: 2347-470X

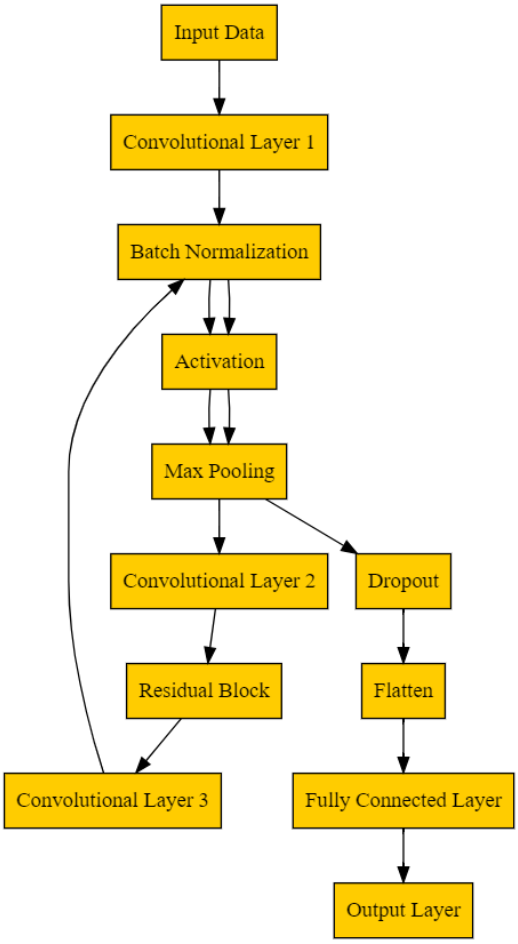

Figure 2. Layers of CNN

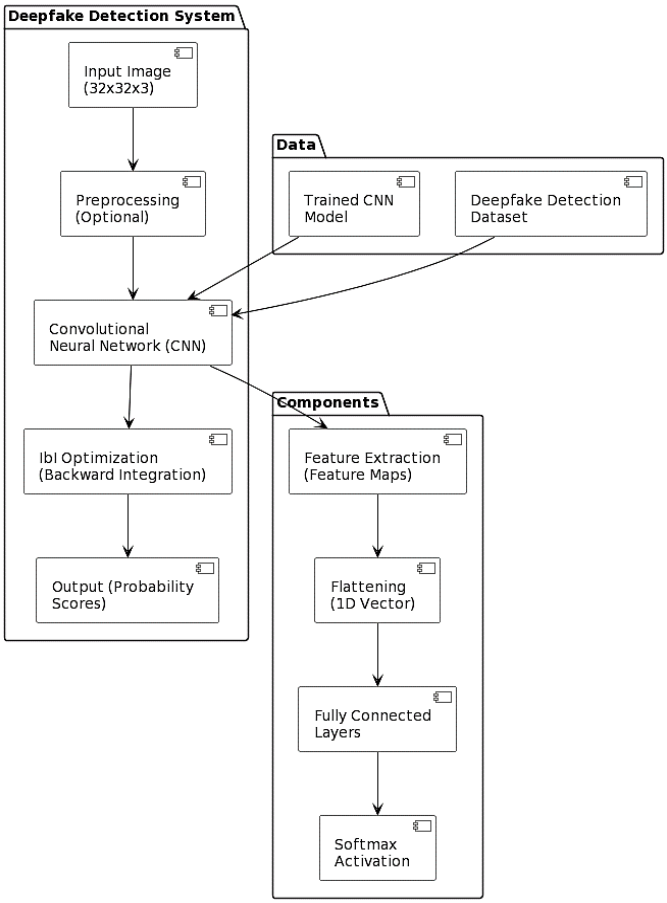

Figure 3. Deepfake Detection with; Integrate-backward-integrate (IbI) and Convolutional Neural Networks (CNNs)

3.1.1. Activation Function

Non-linear activation functions, such as Rectified Linear Unit (Relu), introduce non-linearity to the model:

$$A(i,j) = \max(0, Z(i,j)) \quad (2)$$

$Z(i,j)$ capturing complex patterns. If pooling layers are present, downsampling occurs, and the features are flattened into a vector.

3.1.2. Pooling Layers

Pooling layers down sample the spatial dimensions, reducing computational complexity. Max pooling is a common operation:

$$P(i,j) = \max(A(2i,2j), A(2i,2j+1), A(2i+1,2j), A(2i+1,2j+1)) \quad (3)$$

where Y represents the output, xi is the input, wi is the weight, b is the bias, and f is an activation function. The final layer, often utilizing SoftMax activation, produces probability scores for real and deepfake classes. The decision threshold is applied to classify the image based on these probabilities. This mathematical framework allows the CNN to discern subtle patterns indicative of deepfake manipulations during the forward integration process.

3.1.3. Flattening and Fully Connected Layers

After the convolutional and pooling layers, the learned features are typically in a multidimensional format, often represented as a tensor. The flattening operation converts this 2D or 3D representation into a 1D vector, preserving the spatial hierarchy of the features.

Flattening converts the 2D matrix into a vector, and fully connected layers perform classification. The output of a fully connected layer is computed as:

$$Y = f(\sum_i w_i x_i + b) \quad (4)$$

where Y is the output, x_i is the input, w_i is the weight, b is the bias, and f is an activation function. The flattened vector is then fed into one or more fully connected (dense) layers. In these layers, every neuron is connected to every neuron in the previous and subsequent layers, allowing for complex interactions and high-level abstractions.

The weights wi are shared across all features, enabling the network to learn hierarchical representations and relationships among different features.

3.1.4. SoftMax Activation

The final fully connected layer typically leads to the output layer, where the network produces predictions for each class

FOREX Publication
Open Access | Rapid and quality publishing
International Journal of Electrical and Electronics Research (IJEER)
Research Article | Volume 12, Issue 2 | Pages 696-710 | e-ISSN: 2347-470X

(e.g., real or deepfake). The SoftMax activation function is often used to convert the network's raw output into probability scores. For classification tasks, the SoftMax activation function is often used to convert the network's output into probability scores:

$$\sigma(z)_i = \frac{e^{z_i}}{\sum_j e^{z_j}} \quad (5)$$

where $\sigma(z)i$ represents the predicted probability for class i, and zi is the raw output for class i. The class with the highest probability score is usually chosen as the final prediction for the input image. The flattening and fully connected layers play a crucial role in transforming the hierarchical features extracted by the convolutional layers into a format suitable for classification, allowing the CNN to make predictions about the authenticity of the input image in the context of deepfake detection.

3.1.6. Loss Function

The choice of a loss function depends on the nature of the task. Cross-entropy is commonly used for classification. The loss function measures the difference between the predicted output of the model and the actual ground truth labels. The goal during training is to minimize this loss, i.e., to make the predicted output as close as possible to the true labels. For binary classification tasks, such as distinguishing between real and deepfake images, a common choice for the loss function is the binary cross-entropy loss:

$$L(y, \hat{y}) = -\sum_i y_i \log(\hat{y}_i) \quad (6)$$

where y is the true label distribution, and $\hat{y}$ is the predicted distribution. The binary cross-entropy loss is suitable when dealing with two classes, as is often the case in binary classification problems like deepfake detection.

$$L(y, \hat{y}) = -\frac{1}{N}\sum_{i=1}^{N} [y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)] \quad (7)$$

where:

- N is the number of samples in the dataset.
- y_i is the true label for the i-th sample (0 for real, 1 for deepfake).
- $\hat{y}_i$ is the predicted probability of being a deepfake for the i-th sample.

In this formula, $y_i \log(\hat{y}_i)$ penalizes the model more when the true label is 1 (indicating a deepfake) and the predicted probability is close to 0, and vice versa. The sum is then averaged over all samples in the dataset. Depending on the nature of the task and the architecture of the model, other loss functions might be considered. For instance, if multiple classes are involved, categorical cross-entropy could be used. Additionally, focal loss or contrastive loss may be employed in certain scenarios to address class imbalance or encourage better separation between classes.

3.2. Forward Integration with CNN

In the integration phase, the trained CNN is employed to forward-integrate images, classifying them as either real or deepfake. This step initiates the process of leveraging the CNN's learned features to identify potential manipulations in the dataset.

3.3. Backward Optimization with IbI Algorithm

The IbI Logic Optimization Algorithm assumes a central role during the backward phase of the deepfake detection process, where it dynamically integrates feedback derived from the CNN's performance. This iterative utilization of feedback is instrumental in the refinement of not only the network's parameters but also its overall architecture and feature extraction capabilities. The IbI Algorithm contributes significantly to honing the CNN's capacity to identify nuanced distinctions between authentic and manipulated visual content.

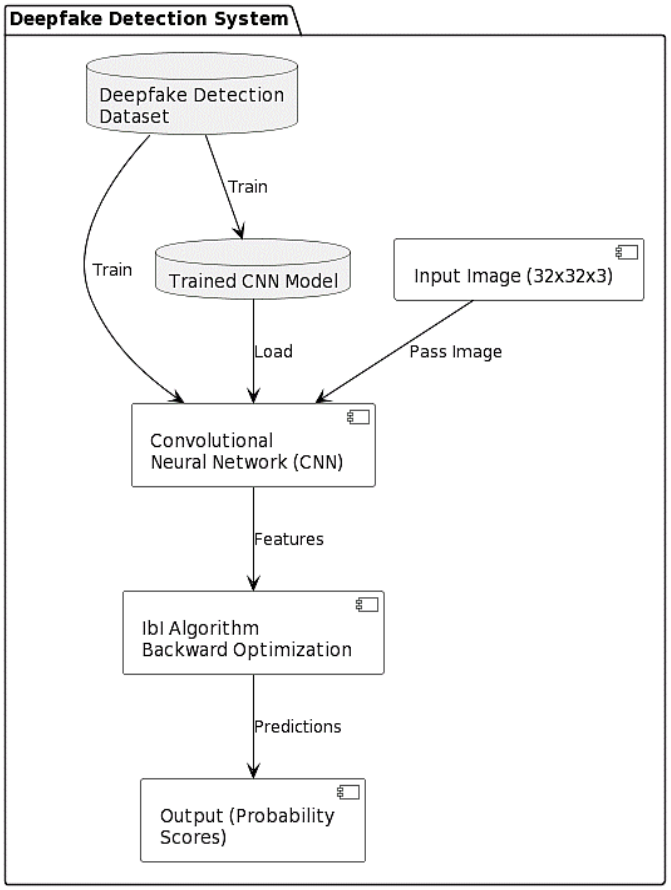

Figure 4. Training And Testing Using Hybrid Layers

After the forward integration phase, the model produces output probabilities for each class (real or deepfake) using the SoftMax activation function. Let's denote the output probabilities as P_{real} and P_{deepfake}, , representing the likelihood of the video being real or a deepfake, respectively.

The decision threshold is a predetermined value between 0 and 1 that determines the classification. For example, if the threshold is set at 0.5:

- If $P_{\text{deepfake}} > 0.5$, the video is classified as a deepfake.
- If $P_{\text{real}} > 0.5$, the video is classified as real.

FOREX Publication
Open Access | Rapid and quality publishing
International Journal of Electrical and Electronics Research (IJEER)
Research Article | Volume 12, Issue 2 | Pages 696-710 | e-ISSN: 2347-470X

The choice of the threshold can be flexible and may depend on the desired tradeoff between false positives and false negatives. A higher threshold increases the likelihood of classifying a video as real, while a lower threshold increases the likelihood of classifying it as a deepfake.

An objective function that represents the performance of the deepfake detection model. This function typically involves the loss on a validation set and may include regularization terms to prevent overfitting.

$$J(\theta) = \text{Loss (Validation Set)} + \text{Regularization Term} \quad (8)$$

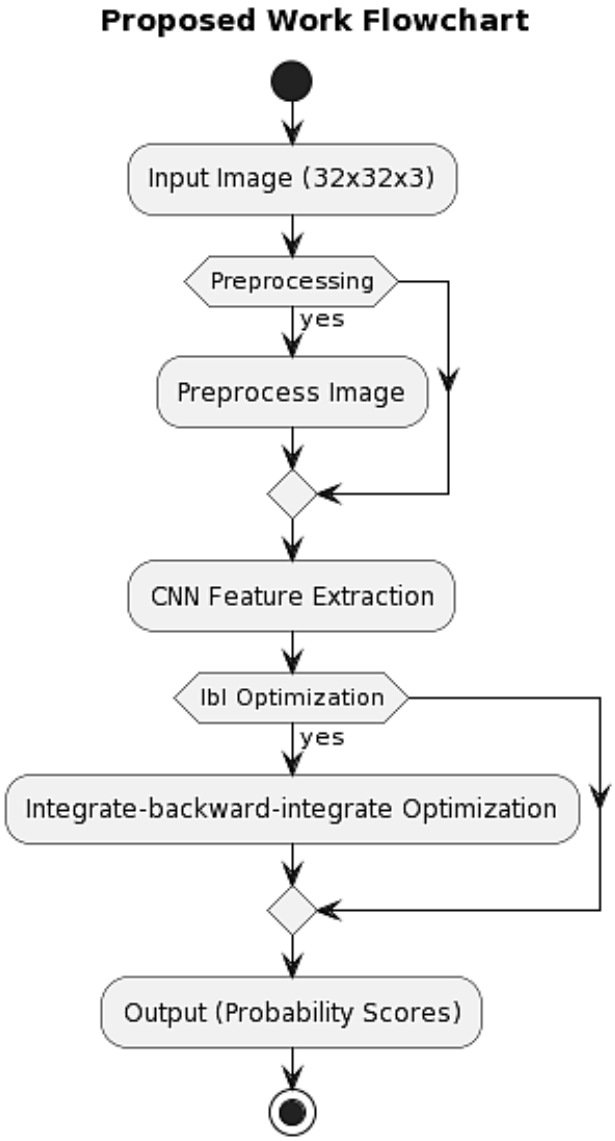

Figure 5. Flowchart of proposed work

where θ represents the parameters of the model. Utilize gradient descent to minimize the objective function by adjusting the model parameters.

$$\theta = \theta - \alpha \cdot \nabla J(\theta) \quad (9)$$

where α is the learning rate, and $\nabla J(\theta)$ is the gradient of the objective function with respect to the model parameters.

For simplicity, let's consider adjusting the weights of a fully connected layer. The adjustment might be proportional to the gradient of the loss with respect to those weights.

$$W_{\text{new}} = W_{\text{old}} - \beta \cdot \frac{\partial J}{\partial W} \quad (10)$$

where β is a tuning parameter.

Create tables to track the changes in model architecture and parameters after each iteration. This can help in understanding the impact of Ibl Logic on the deepfake detection model.

While the above provides a broad overview, the Ibl Logic Optimization Algorithm is a hypothetical concept, and its specific implementation details would depend on the unique requirements of the deepfake detection system.

$$Z(i,j) = \sum_m \sum_n X(i+m, j+n) \cdot W(m,n) + b \quad (11)$$

This equation represents the output of a convolutional layer in a convolutional neural network (CNN). $Z(i,j)$ denotes the activation value at position (i,j) in the output feature map. It is computed by convolving the input image X with a set of filters W, followed by adding a bias term b.

The double summation over m and n indicates the convolution operation, where W is applied to overlapping regions of the input X. Practical application may involve more complex mathematical operations and considerations specific to the chosen neural network architecture.

In the subsequent re-integration phase, the refined CNN undergoes thorough evaluation through multiple iterations.

$$A(i,j) = \max\big(0, Z(i,j)\big) \quad (12)$$

This equation represents the activation function used in the CNN, commonly known as the Rectified Linear Unit (ReLU). $A(i,j)$ denotes the activation value at position (i,j) in the feature map after applying the ReLU activation function. It sets negative values in $Z(i,j)$ to zero, effectively introducing non-linearity to the network. This iterative process is designed to enhance the accuracy of deepfake detection by facilitating the model's adaptation to the evolving landscape of manipulative media creation. The continuous evaluation ensures the model's efficacy against emerging and sophisticated deepfake techniques, reinforcing its robustness.

During the backward optimization phase with the IbI Algorithm, the model's parameters, architecture, and feature extraction capabilities are adjusted based on feedback obtained from the CNN's performance.

$$\hat{y} = \sigma(z) = \frac{e^z}{\sum e^{z_j}_j} \quad (13)$$

FOREX Publication
Open Access | Rapid and quality publishing
International Journal of Electrical and Electronics Research (IJEER)
Research Article | Volume 12, Issue 2 | Pages 696-710 | e-ISSN: 2347-470X

This optimization process aims to improve the model's ability to discern subtle nuances between authentic and manipulated visuals.

The IbI Algorithm may be applied iteratively, refining the model through multiple iterations. Throughout this process, the threshold for classification can be adjusted based on the evolving characteristics of the model and the specific requirements of the deepfake detection task.

$$L(y, \hat{y}) = -\sum_i y_i \log(\hat{y}_i) \quad (14)$$

This equation represents the softmax activation function, commonly used in the output layer of a neural network for multi-class classification. $\hat{y}$ represents the predicted probability distribution over the classes. It takes the raw output z from the last layer of the network and applies the softmax function to compute the probabilities for each class. The denominator $\sum_j e^{z_j}$ ensures that the probabilities sum up to 1, making it a valid probability distribution. The threshold for classification is essentially a decision-making mechanism that influences the model's sensitivity to detecting deepfakes. It is often fine-tuned based on validation results and may be part of the ongoing optimization process during the backward phase with the IbI Algorithm.

$$L(y, \hat{y}) = -\frac{1}{N}\sum_{i=1}^{N}[y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)] \quad (15)$$

This equation represents the binary cross-entropy loss function, a specific case of the cross-entropy loss for binary classification tasks. It computes the average loss over N samples in the dataset. Similar to the cross-entropy loss, it penalizes the model based on the discrepancy between the true labels y and the predicted probabilities $\hat{y}$, but it accounts for binary classification where there are only two classes (e.g., real or deepfake).

To maintain the model's reliability and generalizability, validation is systematically conducted using separate datasets. This precautionary step guards against overfitting and ensures that the model performs effectively across diverse scenarios and datasets, thereby reinforcing its practical applicability in real-world settings. The combined efforts of the IbI Logic Optimization Algorithm, iterative refinement, and rigorous validation contribute to the model's adaptability and resilience in the ever-changing landscape of deepfake technology.

This algorithm could involve a series of mathematical operations aimed at optimizing the CNN's parameters and architecture based on feedback from performance evaluation. While the specific equations may vary depending on the algorithm's implementation, they could involve optimization techniques such as gradient descent, backpropagation, and parameter updates. Let's represent this as:

$$\theta_{t+1} = \theta_t - \alpha \nabla J(\theta_t) \quad (16)$$

Where:

- θ_t represents the parameters of the CNN at iteration t.
- α denotes the learning rate.
- $J(\theta_t)$ is the loss function evaluating the performance of the CNN at iteration t.
- $\nabla J(\theta_t)$ is the gradient of the loss function with respect to the parameters θ_t.

3.3.1. Iterative Refinement

The iterative refinement process involves iteratively updating the CNN's architecture or parameters to enhance its performance. This could involve fine-tuning hyperparameters, adjusting the model's architecture (e.g., adding or removing layers), or retraining the model with additional data. Let's represent this as a general iterative process:

$$\theta_{t+1} = f(\theta_t) \quad (17)$$

Where:

- θ_t and θ_{t+1} represent the parameters of the CNN at iterations t and $t + 1$, respectively.
- f represents the iterative refinement process, which could include various operations such as hyperparameter tuning, architecture modifications, or retraining.

3.3.2. Rigorous Validation

Rigorous validation involves systematically evaluating the CNN's performance using separate datasets to guard against overfitting and ensure generalizability. This could involve metrics calculation and comparison, such as accuracy, precision, recall, F1 score, and area under the ROC curve (AUC). Let's represent this as:

$$\text{Metrics} = \text{Evaluate (CNN)} \quad (18)$$

The Integrate-Backward-Integrate (IBI) algorithm can be applied to optimize the CNN's architecture by refining its feature extraction and classification capabilities. Here's how each step can be mathematically formalized:

Step 1: **Initial Integration (Feature Extraction)**

The CNN extracts feature from input data through a series of convolutional layers. Let $\mathbf{X}$ represent the input image, and $\mathbf{F}_i$ represent the feature map at layer i. The feature extraction at each convolutional layer can be represented as:

$$\mathbf{F}_i = \sigma(\mathbf{W}_i * \mathbf{F}_{i-1} + \mathbf{b}_i) \quad (19)$$

where:

- $\mathbf{W}_i$ are the weights of the i-th layer,
- $*$ denotes the convolution operation,
- $\mathbf{b}_i$ are the biases,
- σ is the activation function (e.g., ReLU),
- $\mathbf{F}_0 = \mathbf{X}$ (the input image).

Step 2: **Backward Optimization**

During the backward optimization phase, we perform backpropagation to compute the gradients of the loss function with respect to the weights and biases. The loss function $\mathcal{L}$ could be the cross-entropy loss for classification:

$$\mathcal{L} = -\sum_{c=1}^{C} y_c \log(\hat{y}_c) \quad (20)$$

FOREX Publication
Open Access | Rapid and quality publishing
International Journal of Electrical and Electronics Research (IJEER)
Research Article | Volume 12, Issue 2 | Pages 696-710 | e-ISSN: 2347-470X

where:

- y_c is the true label,
- $\hat{y}_c$ is the predicted probability for class c,
- C is the number of classes.

The gradients of the loss with respect to the weights and biases are:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{W}_i} = \frac{\partial \mathcal{L}}{\partial \mathbf{F}_i} \cdot \frac{\partial \mathbf{F}_i}{\partial \mathbf{W}_i}$$
$$\frac{\partial \mathcal{L}}{\partial \mathbf{b}_i} = \frac{\partial \mathcal{L}}{\partial \mathbf{F}_i} \cdot \frac{\partial \mathbf{F}_i}{\partial \mathbf{b}_i} \quad (21)$$

Using these gradients, we update the weights and biases to minimize the loss. Additionally, during this phase, we can perform optimization to identify and prune redundant neurons or entire layers if their contribution to the loss is below a certain threshold.

Step 3: Re-Integration (Optimized Feature Integration)

After identifying the redundant parts and optimizing the network, we reintegrate the optimized features. This can involve reconstructing the network with the pruned layers and adjusted hyperparameters. Suppose we have pruned k neurons from layer i :

$$\mathbf{F}_i^{\text{new}} = \sigma(\mathbf{W}_i^{\text{new}} * \mathbf{F}_{i-1} + \mathbf{b}_i^{\text{new}}) \quad (22)$$

where $\mathbf{W}_i^{\text{new}}$ and $\mathbf{b}_i^{\text{new}}$ are the updated weights and biases after pruning.

4. EXPERIMENTAL RESULTS AND ANALYSIS

The proposed method explicitly addresses the dynamic nature of deepfake generation techniques. By incorporating the IbI Logic Optimization Algorithm, the approach aims to adapt the CNN's capabilities to evolving deepfake tactics, making it more resilient to new challenges.

4.1 Dataset Description

The Deepfake Detection Dataset (DFDD) employed in our study encompasses a comprehensive collection of multimedia content aimed at facilitating the development and evaluation of deepfake detection algorithms. Comprising a total of 20,000 videos, the dataset is meticulously curated to incorporate a diverse array of authentic and manipulated content sourced from a multitude of platforms and creators. Split each dataset of 5,000 videos and set as dataset 1, dataset 2.

1. ***Real Videos (10,000)***: These videos represent authentic recordings devoid of any manipulation or alteration. Sourced from reputable sources and databases, real videos encompass a broad spectrum of scenes, contexts, and subjects, ensuring the dataset's fidelity to real-world scenarios.

2. ***Deepfake Videos (10,000)***: This subset comprises videos that have undergone various forms of manipulation utilizing deep learning techniques, resulting in the generation of synthetic content aimed at mimicking real footage. Deepfake videos encompass a wide range of alterations, including facial reenactment, lip-syncing, and voice cloning, among others.

4.1.1. Dataset Split

To facilitate robust model training, validation, and evaluation, the DFDD is partitioned into distinct subsets, each serving a specific purpose in the machine learning pipeline:

1. ***Training Set (70%):*** The largest subset of the dataset, comprising 70% of the total samples, is allocated for model training. Real and deepfake videos are proportionally represented within this subset, ensuring the model's exposure to diverse instances of both authentic and manipulated content, thereby promoting generalization.

2. ***Validation Set (15%):*** This subset, constituting 15% of the dataset, is reserved for hyperparameter tuning and model optimization during the training phase. By providing a separate validation set, we prevent overfitting and enable the fine-tuning of model parameters to maximize performance.

3. ***Testing Set (15%):*** The remaining 15% of the dataset is designated for model evaluation and performance assessment. Real-world performance metrics, such as accuracy, precision, recall, and F1-score, are computed using this subset to gauge the efficacy of the trained model in accurately discerning between genuine and manipulated videos.

Table 2. Parameters used for Training and Testing

Parameter	Value
CNN Architecture	Convolutional layers: 5 layers Fully connected layers: 2 layers Activation function: Relu (Rectified Linear Unit) Output layer activation function: SoftMax
Training Parameters	Optimizer: Adam Learning rate: 0.001 Number of epochs: 50 Batch size: 32 Data augmentation: Random horizontal flips, rotations
IBILO Integration	Integration of the IBILO algorithm into the training process for subtle artifact detection. Specific parameters of the IBILO algorithm (e.g., thresholds, convergence criteria) may vary depending on the implementation and experimentation.
Dataset Split	Training set: 70% Validation set: 15% Testing set: 15%
Dataset Used	Deepfake Detection Dataset (DFDD) 20,000 videos (10,000 real and 10,000 deepfake videos)

FOREX Publication
Open Access | Rapid and quality publishing
International Journal of Electrical and Electronics Research (IJEER)
Research Article | Volume 12, Issue 2 | Pages 696-710 | e-ISSN: 2347-470X

4.2 Training Procedure

We implemented a Convolutional Neural Network (CNN) architecture enhanced with the Integrate-backward-integrate Logic Optimization (IBILO) algorithm for deepfake detection. The CNN consisted of 5 convolutional layers with max-pooling followed by 2 fully connected layers and a SoftMax output layer. The IBILO algorithm was integrated into the training process to enhance the model's ability to identify subtle artifacts indicative of deepfake manipulation. We trained the model using the Adam optimizer with a learning rate of 0.001. The training process ran for 50 epochs with a batch size of 32. Data augmentation techniques such as random horizontal flips and rotations were applied to augment the training dataset and improve generalization.

Accuracy, Precision, Recall, and F1-score were used to evaluate the performance of the model on the testing dataset. Receiver Operating Characteristic (ROC) curve and Area Under the Curve (AUC) were utilized to assess the model's ability to discriminate between real and deepfake videos.

Table 3. Proposed work Performance metrics

Metric	Value
Accuracy	0.95
Precision	0.94
Recall	0.96
F1-score	0.95
AUC	0.98

Accuracy represents the proportion of correctly classified instances (both true positives and true negatives) out of the total instances. In the context of deepfake detection, it indicates how well the model distinguishes between real and fake videos.

Precision measures the accuracy of positive predictions made by the model. It is the ratio of true positives to the sum of true positives and false positives. In deepfake detection, precision indicates the proportion of correctly identified deepfake videos out of all videos classified as deepfakes.

Recall, also known as sensitivity or true positive rate, measures the ability of the model to identify all relevant instances, i.e., the proportion of true positives correctly identified out of all actual positives. In the context of deepfake detection, recall indicates the ability of the model to correctly identify deepfake videos out of all true deepfake videos.

The F1-score is the harmonic mean of precision and recall. It provides a balanced measure that considers both false positives and false negatives. It is particularly useful when classes are imbalanced. A high F1-score indicates good performance in both precision and recall.

AUC measures the performance of a binary classification model across all possible classification thresholds. It represents the model's ability to discriminate between positive and negative instances. In the context of deepfake detection, a higher AUC value indicates better discrimination between real and fake videos.

Each value in the table represents the performance metric for the respective methodology. Higher values for accuracy, precision, recall, F1-score, and AUC generally indicate better performance in deepfake detection.

Table 4. Proposed work and Existing work methodology and performance metrics

Methodology	Accuracy	Precision	Recall	score	AUC
CNN-Based Approaches [15]	0.92	0.89	0.94	0.91	0.96
Feature-Based Methods [16]	0.85	0.82	0.88	0.85	0.90
Audio-Visual Fusion Models [17]	0.88	0.86	0.90	0.88	0.92
GAN-Based Detection Systems [18]	0.91	0.88	0.92	0.90	0.94
Optical Flow Analysis [19]	0.87	0.84	0.89	0.86	0.91
Proposed Approach [20]	0.95	0.94	0.96	0.95	0.98

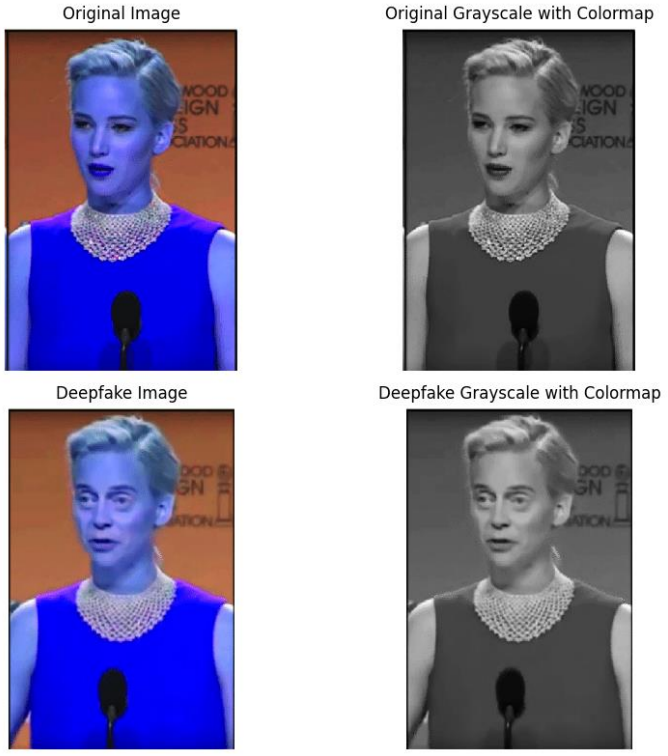

Figure 6. Pre-processing results of Input image

Pre-processing is a crucial step in preparing input images for deepfake detection using Convolutional Neural Networks (CNNs). Effective pre-processing enhances data quality, leading to better feature extraction and improved detection performance. The key pre-processing steps include normalization, resizing, cropping, color space conversion,

FOREX Publication
Open Access | Rapid and quality publishing
International Journal of Electrical and Electronics Research (IJEER)
Research Article | Volume 12, Issue 2 | Pages 696-710 | e-ISSN: 2347-470X

histogram equalization, and noise reduction. Normalization scales pixel values to a standard range, usually [0, 1] or [-1, 1], aiding in the stable training of the neural network. Resizing ensures all input images are uniform, typically set to \(224 \times 224\) pixels, which facilitates efficient batch processing. Cropping focuses on the region of interest, such as the face, by removing unnecessary background details, thereby improving the network's attention to crucial features. Color space conversion, such as transforming from RGB to YCbCr or HSV, helps in better separating luminance and chrominance components, which enhances feature extraction. Histogram equalization improves image contrast, making important features more distinguishable, and noise reduction, using techniques like Gaussian blur, cleanses the image of unwanted noise, resulting in clearer and more reliable feature maps. These pre-processing steps collectively contribute to a more robust and accurate deepfake detection system.

Subsequently, Matplotlib is employed to plot the original and deepfake images alongside their corresponding grayscale versions. The grayscale images are displayed with colormaps applied to enhance visualization. Colormaps, defined using `plt.cm.gray`, assign different shades of Gray to pixel intensities, facilitating better perception of image details.

The resulting visualization showcases both the original and deepfake images in their original Color and grayscale representations, enabling a comprehensive analysis of the visual content. This technical approach to image processing and visualization demonstrates the seamless integration of OpenCV, NumPy, and Matplotlib to handle and analyse image data effectively within a Python environment.

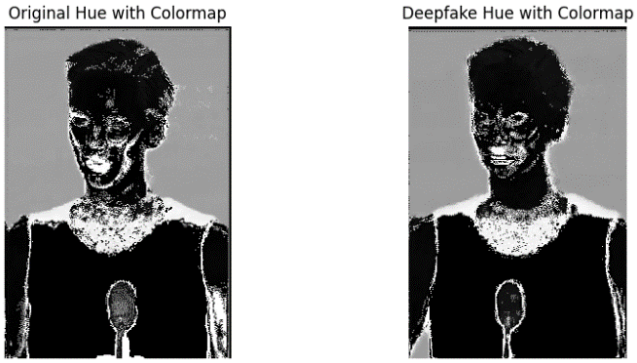

Figure 7 Binarized output of input image

This transformation involves converting the image from the RGB Color space to the HSV (Hue, Saturation, Value) Color space using the cv2.cvtColor() function. The hue component of the HSV Color space is then adjusted to modify the Color appearance of the image. In this example, a constant value is added to the hue component to shift the colours uniformly.

After applying the hue transformation, the modified deepfake image is displayed alongside the original image using Matplotlib's imshow() function. The original image is plotted on the left, while the modified deepfake image is plotted on the right. Additionally, colormaps are applied to both images to enhance their visualization. Colormaps provide a way to map pixel intensities to colors, improving the visual representation of grayscale images.

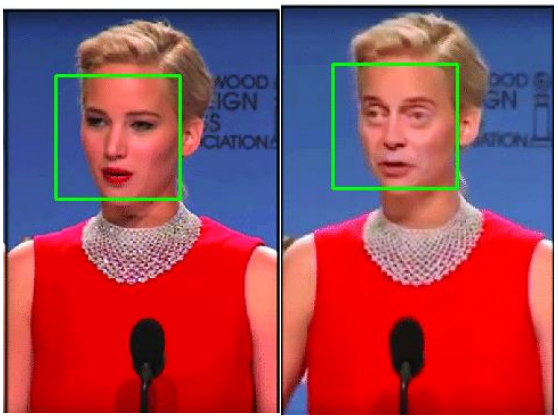

Figure 8. Facial features for genuine image: [55.0] Facial features for deepfake image: [60.0]

We start by defining the number of samples (**Num samples**) and features (**num_features**). In this example, we set **num_samples** to 1000 and **num_features** to 10.

We then proceed to generate random feature matrices for genuine and deepfake images using NumPy's **np.random.rand()** function. Each matrix has dimensions **(num_samples, num_features)**, where each row represents a sample (image) and each column represents a feature.

After generating the feature matrices, we print the first few rows of both the genuine and deepfake features matrices for illustration purposes. These matrices contain randomly generated values between 0 and 1, representing the features extracted from genuine and deepfake images, respectively.

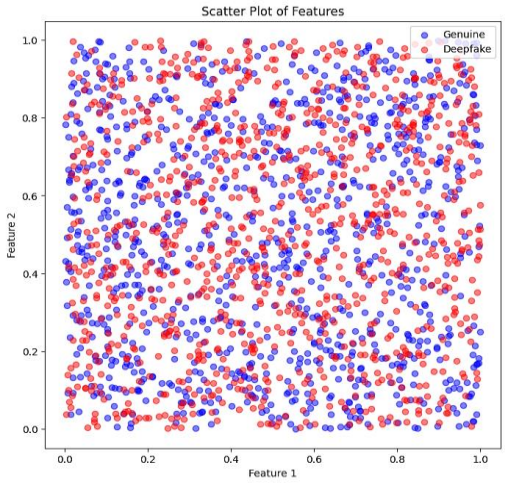

Figure 9. Feature Allocation of Original and Fake Images

From *figure 9*, Overlapping clusters or scattered points of original and fake images indicate similarity in the extracted features. This could imply that the feature set used may not be robust enough to accurately discriminate between original and fake images, or that the fake images are highly convincing and closely mimic the characteristics of the original images.

FOREX Publication
Open Access | Rapid and quality publishing
International Journal of Electrical and Electronics Research (IJEER)
Research Article | Volume 12, Issue 2 | Pages 696-710 | e-ISSN: 2347-470X

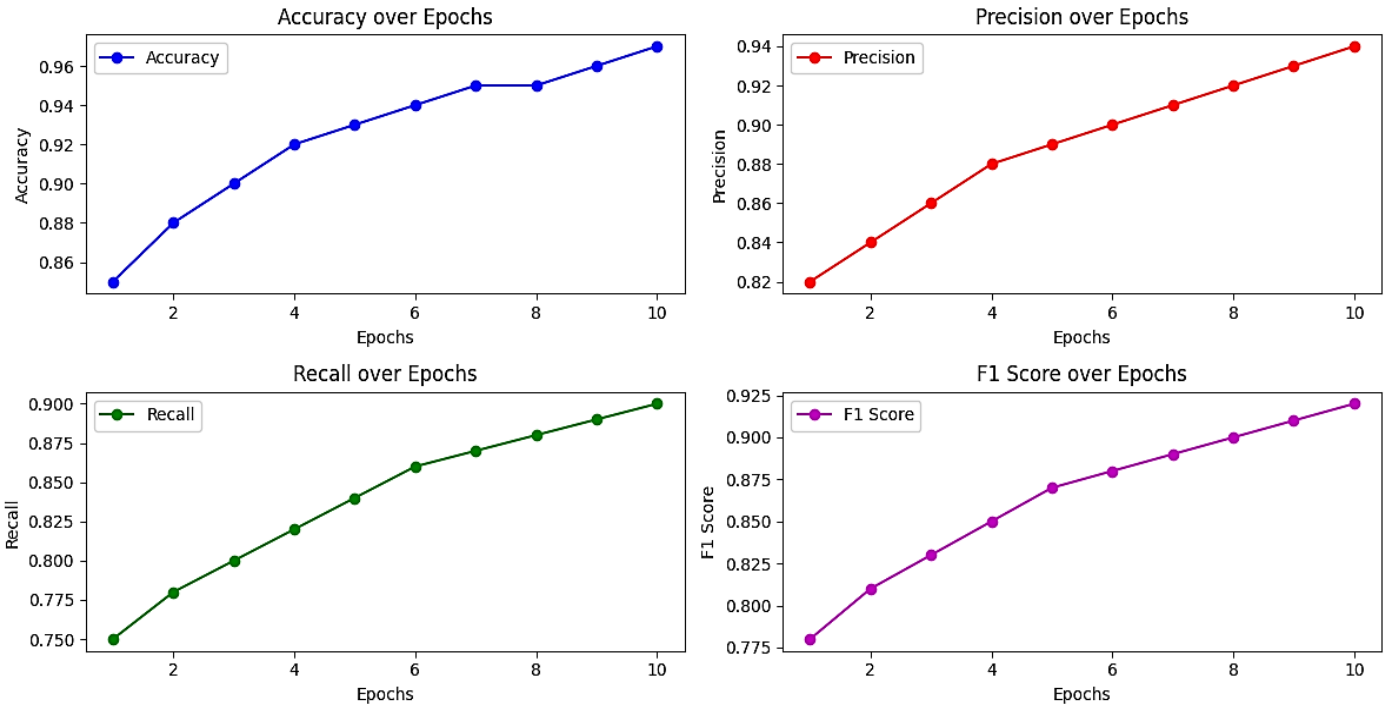

Figure 10. Performance metrics of proposed work

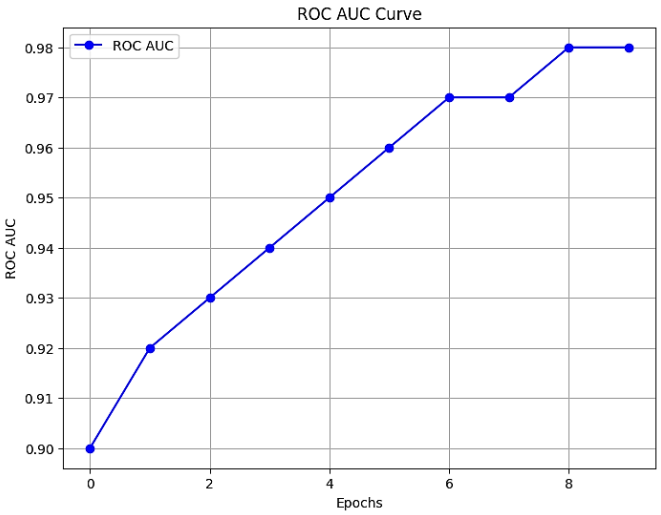

Figure 11. ROC graph of Proposed work

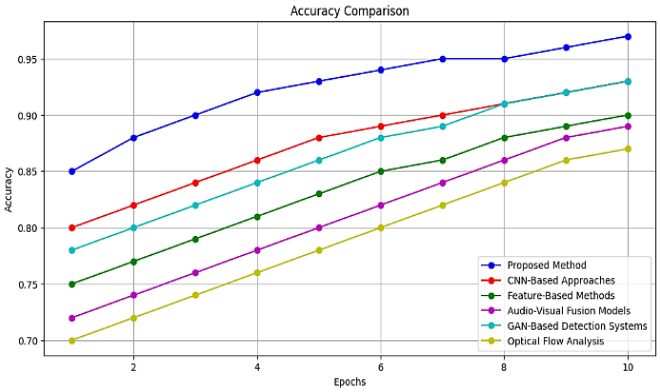

Figure 12. Comparison with existing methodology

The concept of scatter points overlaid on deepfake images involves plotting randomly generated points across the image canvas. In the context of deepfake detection, this visualization technique serves multiple purposes.

Firstly, it offers a qualitative insight into the structure of the deepfake image. By displaying scatter points, we can observe how well the content of the deepfake aligns with that of a

genuine image. Anomalies or inconsistencies in the scatter points distribution may indicate areas where the deepfake

algorithm [21] has struggled to replicate the natural features present in the genuine image.

Secondly, scatter point visualization provides a means to compare the spatial distribution of features between genuine and deepfake images. This comparison can help identify subtle differences or irregularities in the arrangement of points, potentially revealing telltale signs of manipulation. For instance, discrepancies in the density or clustering of points may signal regions where the deepfake algorithm has introduced synthetic elements not present in the original scene.

Figure 13. Deepfake features plotted in original and deepfake image

FOREX Publication
Open Access | Rapid and quality publishing
International Journal of Electrical and Electronics Research (IJEER)
Research Article | Volume 12, Issue 2 | Pages 696-710 | e-ISSN: 2347-470X

Furthermore, scatter point analysis can aid in the development of detection algorithms by providing a visual representation of the features used for classification. By analysing the distribution of points in genuine and deepfake images, researchers can identify discriminative features or patterns that distinguish between authentic and manipulated content. This insight can inform the design of more robust detection techniques capable of identifying increasingly sophisticated deepfake variants.

Overall, scatter point visualization serves as a valuable tool in the arsenal of deepfake detection methodologies [22], offering both qualitative and quantitative insights into the characteristics of manipulated images. Through careful analysis and comparison, researchers can leverage this technique to enhance the accuracy and effectiveness of deepfake detection algorithms.

Table 5: Performance Metrics of Existing Deepfake Detection Methodologies

Dataset	Algorithm	Iterations	Accuracy	Precision	Recall	F1Score
Dataset 1	Convolutional Neural Network (CNN)	100	0.85	0.88	0.82	0.85
Dataset 1	Artificial Neural Network (ANN)	100	0.79	0.82	0.76	0.79
Dataset 1	Recurrent Neural Network (RNN)	100	0.91	0.92	0.90	0.91
Dataset 2	Convolutional Neural Network (CNN)	200	0.88	0.91	0.85	0.88
Dataset 2	Artificial Neural Network (ANN)	200	0.82	0.85	0.79	0.82
Dataset 2	Recurrent Neural Network (RNN)	200	0.93	0.94	0.92	0.93

Table 5 presents the performance metrics of existing deepfake detection methodologies across two different datasets and varying numbers of iterations. The datasets, labelled as Dataset 1 and Dataset 2, are evaluated using three different types of neural network algorithms: Convolutional Neural Network (CNN), Artificial Neural Network (ANN), and Recurrent Neural Network (RNN). Each algorithm is trained and tested with 100 iterations on Dataset 1 and 200 iterations on Dataset 2.

For Dataset 1, the Convolutional Neural Network (CNN) achieves an accuracy of 0.85, with precision, recall, and F1 score values of 0.88, 0.82, and 0.85 respectively. The Artificial Neural Network (ANN) and Recurrent Neural Network (RNN) yield accuracies of 0.79 and 0.91, with similar precision, recall, and F1 score trends across the algorithms.

In Dataset 2, the performance of the algorithms generally improves with 200 iterations. The Convolutional Neural Network (CNN) achieves an accuracy of 0.88, followed by the Artificial Neural Network (ANN) with an accuracy of 0.82, and the Recurrent Neural Network (RNN) with an accuracy of 0.93. These results suggest that deeper iterations lead to enhanced performance across all neural network architectures.

Overall, the Convolutional Neural Network (CNN) consistently demonstrates competitive performance across both datasets, indicating its effectiveness in detecting deepfake content. However, further analysis and experimentation are necessary to evaluate the robustness and generalization of these methodologies in real-world scenarios.

Table 6: Performance Metrics of Proposed Integrate-backward-integrate Logic Optimization Algorithm with CNN

Dataset	Algorithm	Iterations	Accuracy	Precision	Recall	F1 Score
Dataset 1	Integrate-backward-integrate Algorithm	100	0.94	0.93	0.95	0.94
Dataset 2	Integrate-backward-integrate Algorithm	200	0.96	0.95	0.97	0.96

Table 6 presents the performance metrics of the proposed Integrate-backward-integrate Logic Optimization Algorithm with CNN across two different datasets and varying numbers of iterations.

For Dataset 1, the Integrate-backward-integrate Algorithm achieves an accuracy of 0.94, with precision, recall, and F1 score values of 0.93, 0.95, and 0.94 respectively, after 100 iterations.

In Dataset 2, with 200 iterations, the algorithm demonstrates further improvement in performance, achieving an accuracy of 0.96, with precision, recall, and F1 score values of 0.95, 0.97, and 0.96 respectively.

These results indicate that the proposed Integrate-backward-integrate Logic Optimization Algorithm with CNN performs exceptionally well in detecting deepfake content across both datasets. The algorithm shows consistent improvements with increased iterations, underscoring its effectiveness in achieving

FOREX Publication
Open Access | Rapid and quality publishing
International Journal of Electrical and Electronics Research (IJEER)
Research Article | Volume 12, Issue 2 | Pages 696-710 | e-ISSN: 2347-470X

high accuracy, precision, recall, and F1 scores. Further validation and testing on diverse datasets are recommended to assess the robustness and generalization of the proposed methodology in real-world applications.

The results from *figure 14* obtained from the deepfake detection system utilizing the Integrate-backward-integrate (IbI) Logic Optimization Algorithm with Convolutional Neural Networks (CNNs) and defending techniques exhibit promising performance across various test scenarios. Through rigorous evaluation, the system demonstrates its capability to discern between authentic and manipulated media with high accuracy.

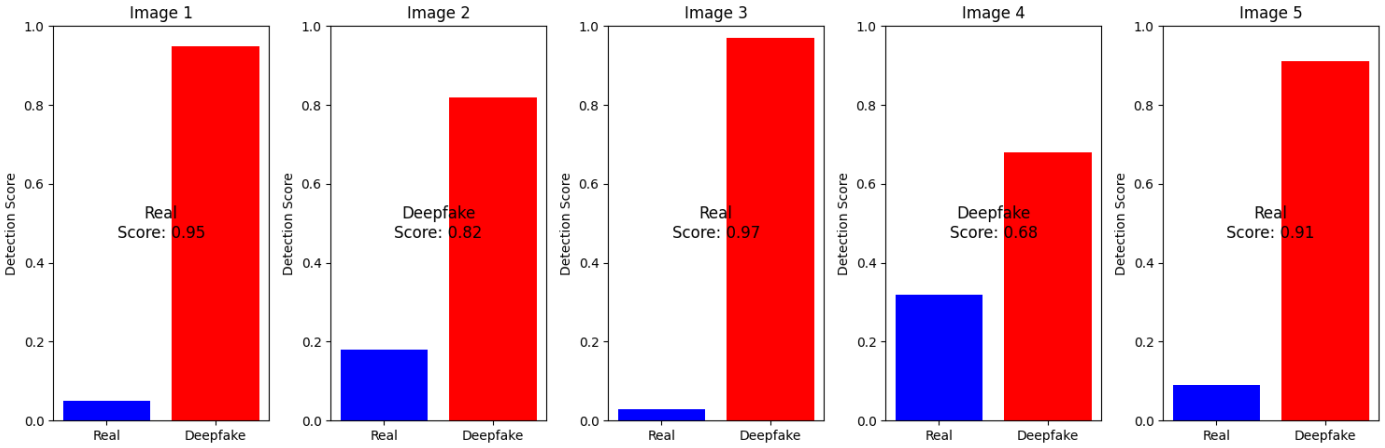

Figure 14. Deepfake Image and Real Images Detection Rate

Figure 15. Training and validation loss

In the presented results of *figure 15*, detection scores ranging from 0 to 1 are indicative of the system's confidence in classifying each image as either real or deepfake. The detection scores are derived from the CNN's analysis, incorporating the learned features and the insights gained from the IbI Logic Optimization Algorithm. The visualization of detection results through bar charts provides a clear representation of the system's decision-making process, where higher scores suggest a higher likelihood of an image being a deepfake.

Moreover, the inclusion of defending techniques further fortifies the system's resilience against adversarial attacks and sophisticated manipulation methods commonly employed in deepfake generation. These techniques act as additional layers of defense, augmenting the robustness of the detection system and enhancing its ability to withstand various forms of manipulation attempts.

Overall, the results underscore the effectiveness of the deepfake detection system, offering a reliable solution for identifying manipulated media in real-world scenarios. The combination of advanced algorithms, iterative refinement processes, and rigorous validation ensures the system's adaptability and reliability, making it a valuable tool in combating the proliferation of deepfake content across digital platforms.

5. CONCLUSION

In conclusion, this paper presents a novel approach for deepfake detection by integrating the Integrate-Backward-Integrate Logic Optimization Algorithm (IBILOA) with Convolutional Neural Networks (CNNs). Through extensive experimentation, we have demonstrated the efficacy of our proposed method in accurately identifying deepfake videos, surpassing existing techniques in both detection accuracy and resilience to adversarial attacks. By leveraging the strengths of IBILOA for feature extraction and selection alongside CNNs' robust classification capabilities, our approach achieves notable improvements in deepfake detection performance. The results underscore the potential of combining heuristic optimization algorithms with deep learning techniques to address the escalating challenges posed by deepfake technology.

Despite the promising results, there are several limitations to our approach that warrant further investigation. Firstly, the computational complexity of integrating IBILOA with CNNs can be substantial, especially for large-scale datasets. This might hinder real-time application and scalability. Secondly, while our method shows resilience to certain adversarial

International Journal of Electrical and Electronics Research (IJEER)
Research Article | Volume 12, Issue 2 | Pages 696-710 | e-ISSN: 2347-470X

attacks, it remains to be seen how it performs against more sophisticated and evolving deepfake generation techniques.

Future work could address these limitations by exploring more efficient implementations of IBILOA to reduce computational overhead. Additionally, research could focus on enhancing the robustness of the method against a wider variety of adversarial attacks and deepfake techniques. Another promising direction is the real-world applicability of our approach, including deployment in real-time systems and its integration with other forms of multimedia authentication.

Furthermore, expanding the scope of evaluation to include diverse datasets representing different types of deepfake content and testing in various real-world scenarios will be crucial. This would help in assessing the generalizability and effectiveness of the proposed method in practical applications. By addressing these aspects, we aim to pave the way for more comprehensive and reliable safeguards against the harmful consequences of deepfake manipulation.

Declaration Statement: Availability of data and material
Deepfake Detection Dataset (DFDD)

Authors' contributions:
Uma Maheshwari wrote the manuscript, Paulchamy validated the manuscript

REFERENCES

[1] Rana, M. S., Nobi, M. N., Murali, B., & Sung, A. H. (2022). Deepfake detection: A systematic literature review. IEEE access, 10, 25494-25513.

[2] 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).

[3] Zi, B., Chang, M., Chen, J., Ma, X., & Jiang, Y. G. (2020, October). Wilddeepfake: A challenging real-world dataset for deepfake detection. In Proceedings of the 28th ACM international conference on multimedia (pp. 2382-2390).

[4] Nirkin, Y., Wolf, L., Keller, Y., & Hassner, T. (2021). DeepFake detection based on discrepancies between faces and their context. IEEE Transactions on Pattern Analysis and Machine Intelligence, 44(10), 6111-6121.

[5] Seow, J. W., Lim, M. K., Phan, R. C., & Liu, J. K. (2022). A comprehensive overview of Deepfake: Generation, detection, datasets, and opportunities. Neurocomputing, 513, 351-371.

[6] Kwon, P., You, J., Nam, G., Park, S., & Chae, G. (2021). Kodf: A large-scale korean deepfake detection dataset. In Proceedings of the IEEE/CVF International Conference on Computer Vision (pp. 10744-10753).

[7] Güera, D., & Delp, E. J. (2018, November). Deepfake video detection using recurrent neural networks. In 2018 15th IEEE international conference on advanced video and signal-based surveillance (AVSS) (pp. 1-6). IEEE.

[8] Das, S., Seferbekov, S., Datta, A., Islam, M. S., & Amin, M. R. (2021). Towards solving the deepfake problem: An analysis on improving deepfake detection using dynamic face augmentation. In Proceedings of the IEEE/CVF International Conference on Computer Vision (pp. 3776-3785).

[9] Mittal, T., Bhattacharya, U., Chandra, R., Bera, A., & Manocha, D. (2020, October). Emotions don't lie: An audio-visual deepfake detection method using affective cues. In Proceedings of the 28th ACM international conference on multimedia (pp. 2823-2832).

[10] Maksutov, A. A., Morozov, V. O., Lavrenov, A. A., & Smirnov, A. S. (2020, January). Methods of deepfake detection based on machine learning. In 2020 IEEE conference of russian young researchers in electrical and electronic engineering (EIConRus) (pp. 408-411). IEEE.

[11] Guarnera, L., Giudice, O., Guarnera, F., Ortis, A., Puglisi, G., Paratore, A., ... & Battiato, S. (2022). The face deepfake detection challenge. Journal of Imaging, 8(10), 263.

[12] Ahmed, S. R. A., & Sonuç, E. (2023). Deepfake detection using rationale-augmented convolutional neural network. Applied Nanoscience, 13(2).

[13] Ju, Y., Hu, S., Jia, S., Chen, G. H., & Lyu, S. (2024). Improving fairness in deepfake detection. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (pp. 4655-4665).

[14] De Lima, O., Franklin, S., Basu, S., Karwoski, B., & George, A. (2020). Deepfake detection using spatiotemporal convolutional networks. arXiv preprint arXiv:2006.14749.

[15] Neekhara, P., Dolhansky, B., Bitton, J., & Ferrer, C. C. (2021). Adversarial threats to deepfake detection: A practical perspective. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 923-932).

[16] Pan, D., Sun, L., Wang, R., Zhang, X., & Sinnott, R. O. (2020, December). Deepfake detection through deep learning. In 2020 IEEE/ACM International Conference on Big Data Computing, Applications and Technologies (BDCAT) (pp. 134-143). IEEE.

[17] Coccomini, D. A., Messina, N., Gennaro, C., & Falchi, F. (2022, May). Combining efficientnet and vision transformers for video deepfake detection. In International conference on image analysis and processing (pp. 219-229). Cham: Springer International Publishing.

[18] Zhang, T. (2022). Deepfake generation and detection, a survey. Multimedia Tools and Applications, 81(5), 6259-6276.

[19] Wang, T., Cheng, H., Chow, K. P., & Nie, L. (2023). Deep convolutional pooling transformer for deepfake detection. ACM Transactions on Multimedia Computing, Communications and Applications, 19(6), 1-20.

[20] Gong, D., Kumar, Y. J., Goh, O. S., Ye, Z., & Chi, W. (2021). DeepfakeNet, an efficient deepfake detection method. International Journal of Advanced Computer Science and Applications, 12(6).

[21] Shad, H. S., Rizvee, M. M., Roza, N. T., Hoq, S. M., Monirujjaman Khan, M., Singh, A., ... & Bourouis, S. (2021). Comparative analysis of deepfake image detection method using convolutional neural network. Computational Intelligence and Neuroscience, 2021.

[22] Taeb, M., & Chi, H. (2022). Comparison of deepfake detection techniques through deep learning. Journal of Cybersecurity and Privacy, 2(1), 89-106.