Detection of Manipulated Multimedia In Digital Forensics Using Machine Learning

Preetam Anvekar
Department of Computer Science and Engineering
Jain College of Engineering and Research
Belagavi, India
preetamanvekar79@gmail.com

Anand Gudnavar*
Department of Computer Science and Engineering
Jain College of Engineering and Research
Belagavi, India
anand_gudnavar@yahoo.co.in

Keerti Naregal
Department of Computer Science and Engineering
Graphic Era University
Dehradun, India
keerti.n@geu.ac.in

Sreedevi Nagarmunoli
Department of Computer Science and Engineering
Graphic Era University
Dehradun, India
sreedevi.n@geu.ac.in

***Abstract*— The surge in digital media usage has spurred an uptick in multimedia manipulation, spanning images, videos, and audio. This manipulation, with its potential to spread misinformation and manipulate public opinion, poses serious threats. Detecting genuine from fake media is challenging due to the diverse tools employed. Consequently, cybercrime involving manipulated media is on the rise. Researchers are countering this issue with machine learning techniques, particularly Convolutional Neural Networks (CNNs). This paper presents an application leveraging CNNs to identify genuine and fake media, bolstered by results from experiments on real and manipulated datasets, yielding high accuracy and robustness. Deep learning models excel in detecting various manipulation types, positioning them as potent weapons against manipulated content proliferation. To ascertain the models' effectiveness, the study includes comprehensive validation, testing procedures, and robustness analyses against sophisticated manipulations, including adversarial attacks and deepfake variations. This research advances multimedia forensics, offering a holistic approach to detect manipulated media with deep learning models, underscoring CNNs' effectiveness in curbing manipulated content dissemination, and emphasizing the necessity of ongoing advancements to tackle the evolving multimedia manipulation landscape.**

Keywords*— *Manipulated media, CNN, Machine learning, Digital forensics, Cyber security.

I. Introduction

The aim to develop a system for detecting manipulated multimedia content using CNNs. The system will focus on detecting manipulations in images, videos, and audio recordings, and will be trained on a large dataset of authentic and manipulated content. The research will explore the latest techniques in deep learning and image processing to develop a system that can accurately classify the authenticity of multimedia content.

The research will begin with a literature review of current state-of-the-art techniques for detecting manipulated multimedia using CNNs [1]. Based on this review, the appropriate CNN architecture will be selected for the system as it is most recently used method [2]. The next step will be to identify the necessary pre-processing steps to prepare the data for analysis, such as data normalization and augmentation. The system will be trained using techniques such as stochastic gradient descent and backpropagation to minimize the error between the predicted and actual values. The performance of the model will be evaluated using metrics viz., as specificity, sensitivity, accuracy, and precision. The system will also be tested on a separate dataset to validate its performance.

The work will address the challenges and limitations of using CNNs for detecting manipulated multimedia. This includes the lack of large, high-quality datasets for training, the difficulty of detecting manipulations, and the potential for adversarial attacks. The project will explore techniques such as MFCC, ELA.

This paper addresses the detection of manipulated multimedia using Machine Learning Techniques. The subsequent sections are structured as follows. Section 2 delves into background of manipulated multimedia detection. Section 3 outlines the proposed methodology, and Section 4 provides details on the experimental setup and results. Finally, Section 5 concludes the paper.

II. Background

The detection of manipulated multimedia using machine learning (ML) has become an increasingly important research area in recent years, driven by the growing concerns about fake news and misinformation, the need for effective forensic tools, and the implications for national security [3]. We will explore the motivations for research in this area, the current state-of-the-art techniques for detecting manipulated multimedia [4], and the challenges and opportunities for future research.

One of the main motivations for research in this area is the increasing prevalence of fake news and misinformation in today's digital world [3]. With the widespread availability of sophisticated image and video editing tools, it has become relatively easy to create doctored content that can mislead or deceive people. This has serious implications for society, as misinformation can fuel social unrest, political instability, and even violence. In this context, the detection of manipulated multimedia using ML can play an important role in mitigating the problem by identifying and flagging fake or doctored content.

Another important motivation for research in this area is the need for more effective forensic tools to detect and authenticate multimedia content. With the rise of deepfakes and other advanced techniques for manipulating multimedia, traditional forensic techniques are becoming less effective [5]. ML-based techniques, on the other hand, have shown promise in detecting manipulated multimedia. For example, researchers have developed algorithms that can identify anomalies in the lighting, shadows, and reflections of images and videos, which can indicate the presence of manipulations. Other researchers have used ML to analyze the patterns of audio to detect whether it is genuine or tampering. These techniques have the potential to be more effective than traditional forensic methods and can help authenticate multimedia content more accurately [6].

III. Proposed Methodology

Training a machine learning model invariably involves navigating the trade-off between overfitting and underfitting, exerting a notable influence on the model's real-time efficacy. Effectively managing this trade-off presents a formidable challenge. A significant concern arises in deep fake scenarios, where the prevalence of a high false-positive rate poses a primary challenge. This occurs when models erroneously label an unfamiliar pattern as abnormal, especially if it wasn't present in the training set. This challenge stems from the inherent difficulty of training the model on a comprehensive dataset that encompasses all conceivable patterns and cases, whether they are deep artificial or genuine. This scenario is often regarded more as a theoretical concept than a practically executable undertaking within the confines of current capabilities [7].

A. *Data Pre-processing*

Data pre-processing plays an important role in detecting manipulated multimedia content using machine learning, especially Convolutional Neural Networks (CNNs) [2]. Pre-processing techniques aim to standardize the input data, improve its quality, and ensure compatibility with CNN models.

Image pre-processing is crucial for compatibility and model performance. This involves resizing images to a consistent size, often 224x224, normalizing pixel values to [0, 1] by dividing by the maximum value (e.g., 255), and applying data augmentation techniques like rotation, flipping, and zooming to enhance dataset diversity and model generalization. Augmentation can also include methods specific to image manipulation, simulating various forms of splicing or retouching.

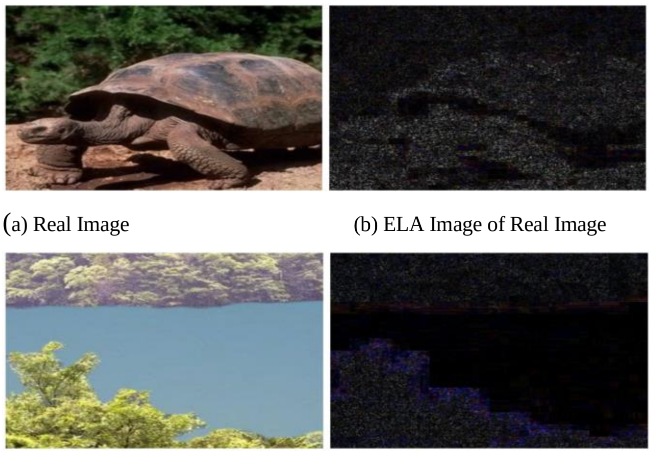
B. *Feature Extraction*

Feature extraction in image media is a fundamental step in various computer vision tasks, including the detection of manipulated multimedia using machine learning techniques such as Convolutional Neural Networks (CNNs). Feature extraction involves transforming raw image data into a more compact and representative representation that captures meaningful information for subsequent analysis and classification.

1) *Image and Video Feature Extraction*

CNNs automatically learn hierarchical features from ELA images by convolving filters across the image and extracting local patterns. The filters in the early layers capture low-level features such as edges and textures, while deeper layers capture more complex and abstract features.

The activations of the CNN layers, such as the output of convolutional or pooling layers, can be considered as extracted features that capture discriminative information about the image.



(a) Real Image (b) ELA Image of Real Image

(c) Fake Image (d) ELA Image of fake Image

Fig. 1. (a) is real image, (b) is ELA of real image and (c) is fake image, (d) is ELA of fake image

ELA function: Fig. 1 (b) and (d) are the ELA image (Error Level Analysis) is a technique used in image forensics to detect potential areas of manipulation or tampering in an image [8]. It compares the error levels introduced during compression and helps identify regions that may have been altered.

i Image Compression:
- Convert the image to a lossy compressed format, such as JPEG, using a specified compression quality factor (QF).

ii Resave the Compressed Image:
- Save the compressed image at a known quality factor (QF') different from the initial compression quality factor.

iii Compute the Difference Image:
- Subtract the pixel values of the resaved image from the original compressed image, generating a difference image.

iv Calculate the Error Level:
- Calculate the error level of each pixel in the difference image by quantifying the differences between the original and resaved image.
- The error level can be computed using various methods, such as Mean Squared Error (MSE) or Absolute Difference.

Mean Squared Error (MSE):

$$\text{MSE} = \Sigma\, ((I(x, y) - I'(x, y))\, \hat{}\, 2) / (W * H) \qquad (1)$$

Where, I(x, y) is the pixel value of the original image at coordinates (x, y). I'(x, y) is the pixel value of the resaved image at coordinates (x, y). W and H are the width and height of the image, respectively.

Absolute Difference:

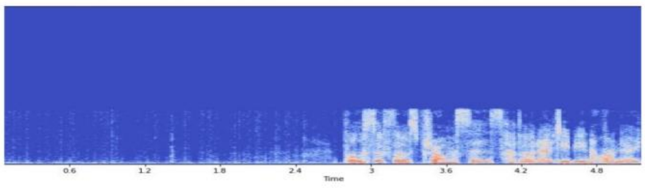
$$\text{Diff}(x, y) = |I(x, y) - I'(x, y)| \qquad (2)$$

Where, I(x, y) is the pixel value of the original image at coordinates (x, y). I'(x, y) is the pixel value of the resaved image at coordinates (x, y).

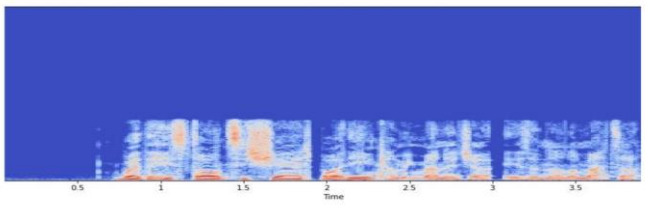
2) Audio Feature Extraction

Fig. 2 (a) and (b) shows the MFCC series audio files [7]. MFCCs are widely used features for audio analysis. They capture the spectral content of an audio signal by converting the power spectrum of the signal into a logarithmic scale that mimics human auditory perception.

The process of computing MFCC encompasses multiple stages. These include employing the Fast Fourier Transform (FFT) to acquire the power spectrum, utilizing the Mel filterbank to extract pertinent frequency bands, logarithmically transforming the filterbank energies, and ultimately employing the Discrete Cosine Transform (DCT) to derive the coefficients.



a. Fake audio



b. Real Audio

Fig. 2. (a) and (b) are image of mel-spectrogram of real and fake audio

MFCC function steps

Preemphasis:

The input audio signal is preemphasized to emphasize higher frequencies and improve the stability of the subsequent analysis.

Preemphasized signal:

x_preemp[n] = x[n] - α* x [n-1] (3)

where, x[n] is the input audio signal and α is the preemphasis coefficient.

Framing:

The preemphasized signal is divided into frames of equal length, typically with a duration of 20-40 milliseconds.

Frame size:

N = sampling_rate * frame_duration (4)

where sampling_rate is the audio sampling rate in Hz and frame_duration is the desired frame duration in seconds.

Number of frames:

K = floor ((N - frame_overlap) / (N - frame_shift)) (5)

Where, frame_overlap is the number of samples overlapping between consecutive frames, and frame_shift is the number of samples shifted for each consecutive frame.

Windowing:

Each frame is multiplied by a window function to reduce spectral leakage caused by the abrupt truncation of the signal.

Windowed frame:

x_windowed[k, n] = x_preemp[(k * frame_shift) + n] * w[n] (6)

where, k is the frame index, n is the sample index within the frame, and w[n] is the window function applied to the frame.

Fast Fourier Transform (FFT):

The windowed frames are transformed into the frequency domain using the Fast Fourier Transform (FFT).

Spectrum of the frame:

X[k, m] = FFT(x_windowed[k, n]) (7)

where, m is the frequency bin index and X[k, m] represents the complex-valued spectrum.

Mel Filterbank:

The Mel filterbank is applied to the power spectrum to extract relevant frequency bands that mimic human auditory perception.

Mel filterbank output:

H[k, i] = sum(|X[k, m]|^2 * H_mel[m, i]) (8)

where i is the filter index and H_mel[m, i] represents the response of the i-th Mel filter at frequency bin m.

Logarithm:

The Mel filterbank outputs are converted to the logarithmic scale to approximate the logarithm of the perceived loudness.

Log Mel filterbank output:

L[k, i] = log(H[k, i]) (9)

Discrete Cosine Transform (DCT):

The Discrete Cosine Transform (DCT) is applied to the log Mel filterbank outputs to decorrelate the coefficients and capture the cepstral features.

MFCC coefficient:

C[k, j] = sum(L[k, i] * cos((pi * j * (I – 0.5)) / num_filters)) (10)

where, j is the MFCC coefficient index and num_filters is the number of Mel filters.

C. Classification Model

VGG16 for image and video detection

Fig. 3 uses VGG16 model to train the images and video for detection of manipulated image and video. VGG16 model is a popular convolutional neural network architecture that consists of 16 layers, including 13 convolutional layers and 3 fully connected layers.

Convolutional Layers:

The VGG-16 model starts with several convolutional layers, where each layer performs convolutions on the input image using a set of learnable filters. The outputs of these layers capture different levels of image features.

The convolutional layers use small receptive fields (3x3) with a stride of 1, and they apply zero-padding to preserve the spatial dimensions of the feature maps.

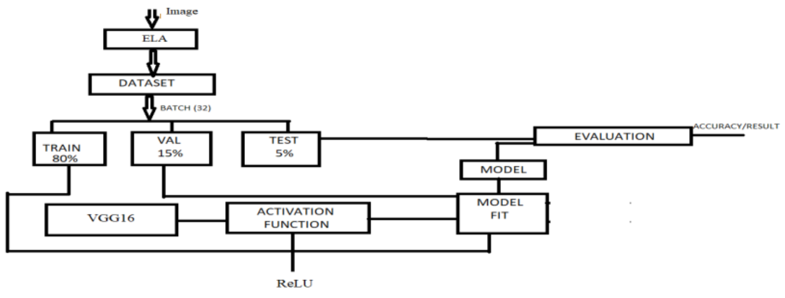


Fig. 3. Model Architecture for Image detection using VGG16

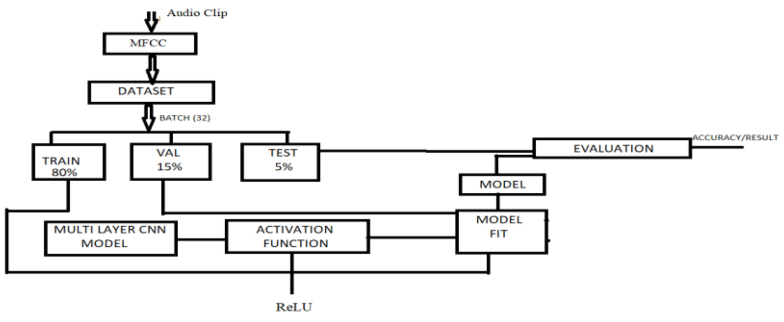


Fig. 4. Model Architecture for Audio detection using CNN

Input: X with dimensions (W_in, H_in, D_in)
Convolution operation:

Z_i = Conv(W_i * X) + b_i (11)

where, * denotes the convolution operation, W_i represents the i-th set of learnable filters, b_i is the corresponding bias term.
Activation function:

A_i = ReLU(Z_i) (12)

which involves applying the rectified linear unit (ReLU) activation function individually to each element of the output derived from the convolution operation.

Output: The output feature map of the i-th convolutional layer has dimensions (W_i_out, H_i_out, D_i_out), where W_i_out and H_i_out depend on the stride and padding parameters, and D_i_out is the number of output channels.

Max Pooling Layers:

After each set of convolutional layers, the VGG-16 model incorporates max pooling layers, which downsample the feature maps by selecting the maximum value within a defined pooling window.

The pooling layers help reduce the spatial dimensions of the feature maps while retaining the most salient features.

Input: A_i with dimensions (W_i_in, H_i_in, D_i_in)

Pooling operation:

P_i = MaxPool(A_i) (13)

where, P_i is the downsampled feature map obtained by selecting the maximum value within a pooling window.

Output: The output feature map of the i-th pooling layer has dimensions (W_i_out, H_i_out, D_i_in), where, W_i_out and H_i_out depend on the pooling window size and stride, and D_i_in remains unchanged.

Fully Connected Layers:

The final layers of the VGG-16 model are fully connected layers, which take flattened feature maps as input and perform classification based on the learned features.

CNN for audio detection

We employ a similar MFCC architecture of Fig.4 for Audio detection. In this architecture multi-layer CNN model [6] is used to extract audio feature from MFCC audio data.

Convolutional Layer:

Input: X with dimensions (W_in, H_in, D_in), where W_in is the width (number of time steps), H_in is the height (number of MFCC coefficients), and D_in is the number of input channels.
Convolution operation:

Z = Conv(X, W) + b (14)

where W represents the learnable filters, b is the corresponding bias term.
Activation function:

A = ReLU(Z) (15)

which involves applying the rectified linear unit (ReLU) activation function individually to each element of the output derived from the convolution operation.
Output: The output feature map has dimensions (W_out, H_out, D_out), where W_out and H_out depend on the stride and padding parameters, and D_out is the number of output channels.

Max Pooling Layer:

Input: A with dimensions (W_in, H_in, D_in).

Pooling operation:

P = MaxPool(A) (16)

where P is the downsampled feature map obtained by selecting the maximum value within a pooling window depend on the pooling window size and stride, and D_in remains unchanged.

Output: The output feature map has dimensions (W_out, H_out, D_in), where W_out and H_out.

Fully Connected Layer:

Input: F with dimensions (N, M), where N is the number of input features and M is the number of neurons in the fully connected layer.
Linear transformation:

$$Z = F * W + b \tag{17}$$

where * denotes the matrix multiplication, W represents the weight matrix, and b is the bias term.
Activation function:

$$A = ReLU(Z) \tag{18}$$

which involves applying the ReLU activation function individually to each element of the output derived from the linear transformation.
Output: The output of the fully connected layer is A with dimensions (N, M).

IV. Experiment And Result

About 12,000 images in dataset used to detect manipulated image and video. Approximately 7,000 are fake images and 5,000 real images. For audio detection about 3,00,000 samples are used. Approximately 1,30,000 are fake audio and 1,70,000 real audio. This is large dataset to training the model.

A. ASVspoof 2019

The ASVspoof 2019 dataset [9] is a benchmark dataset used for the evaluation of automatic speaker verification (ASV) systems in the context of spoofing attacks. It was released as part of the ASVspoof 2019 Challenge organized by the Automatic Speaker Verification Spoofing and Countermeasures (ASVspoof) consortium.

The dataset assesses the vulnerability of ASV (Automatic Speaker Verification) systems to spoofing attacks by incorporating genuine and spoofed speech recordings in multiple languages like English, Mandarin, Cantonese, and Vietnamese, generated using techniques such as speech synthesis, voice conversion, and replay attacks. It introduces acoustic variability through different microphones and recording devices while maintaining high audio fidelity. Furthermore, the dataset is divided into training, development, and evaluation subsets for ASV system training and evaluation, with recordings in standard audio formats, accompanied by metadata detailing recording information, speaker identity, and spoofing method. ASVspoof 2019 serves as a valuable resource for testing spoofing countermeasures and as a benchmark for assessing ASV system resilience against spoofing attacks, contributing to advancements in speaker verification anti-spoofing techniques.

TABLE I. Audio Detection Based System Comparison

	Existing System	Proposed System
Method	Spectrogram	MFCC
Model	CNN	CNN
Training Accuracy	94%	97.34%
Testing Accuracy	89%	96.07%

Table I compare the existing system and proposed system based on some properties such as method and algorithm used and accuracy of training and testing of the model. In existing system method is used is spectrogram which take more time to train and less accuracy, to overcome the time complexity proposed system use the MFCC Scaling which is spectrogram of spectrogram. Fig. 5 shows the accuracy increased for each epoch. Accuracy for 50 epochs is 97%

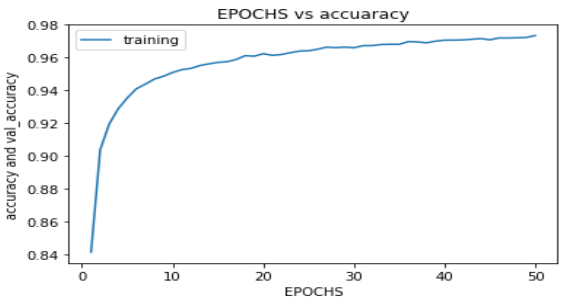


Fig. 5. Graph representation of Training of Audio Detection (EPOCHS versus Accuracy)

B. CASIA2.0

The CASIA Image Tampering Detection Evaluation Database (CASIA TIDE) is a benchmark dataset tailored for image tampering detection research and widely used in digital image forensics. It offers a standardized resource for evaluating the performance of tampering detection techniques by encompassing a diverse collection of digital images subjected to various tampering scenarios, including copy-move, splicing, and removal. Each image in the dataset is provided in common formats like JPEG and comes with ground truth annotations specifying tampering locations and types, making it a valuable asset for tampering detection algorithm development and assessment in the field of digital image forensics.

TABLE II. Image And Video Detection Based System Comparison

	Existing System	Proposed System
Image Size	150X150	224X224
Model	VGG16	VGG16
Training Accuracy	90.97%	92.17%
Testing Accuracy	86.12%	87.87%

Table II compare the existing system and proposed system based on some properties such as image size and algorithm used and accuracy of training and testing of the model. CASIA 2.0 dataset is used for video and image detection in proposed system. Image size in existing system was 150X150 was given with accuracy of 90%, were in proposed system size was changed to 224 X 224 which standard size was produced 92 %. Fig. 6 shows the accuracy increased for each epoch. Accuracy for 10 epochs is 92%.

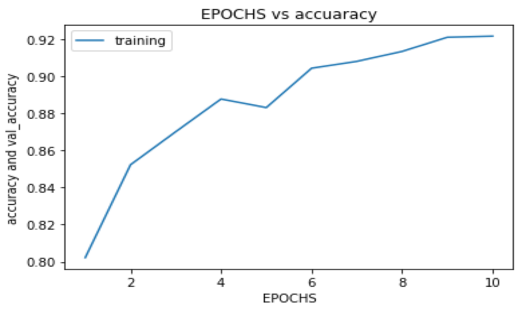


Fig. 6. Graph representation of Training of Image and Video Detection (EPOCHS versus Accuracy)

C. *Results Screen Shots*

Following Figures, Fig. 7 – Fig. 12 depict the actual screen shots of the Application.

Figures show the frontend status during prediction of Real and Fake Audio, Real and Fake Image.

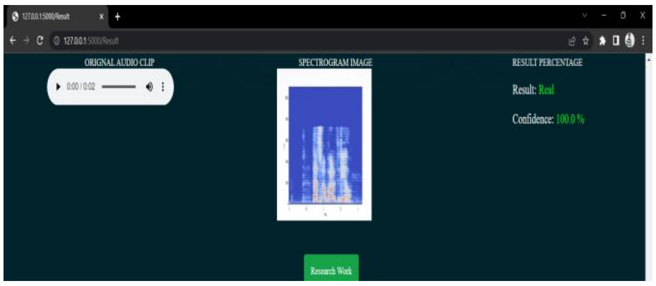


Fig. 7. Prediction of Real Audio

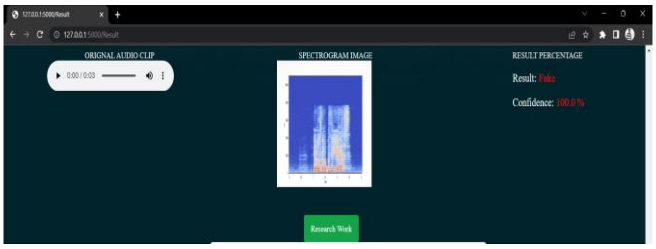


Fig. 8. Prediction of Fake Audio

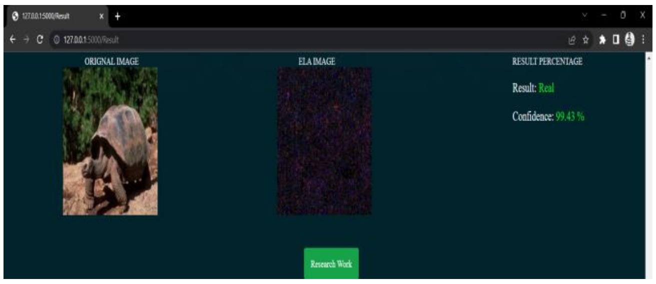


Fig. 9. Prediction of Real Image

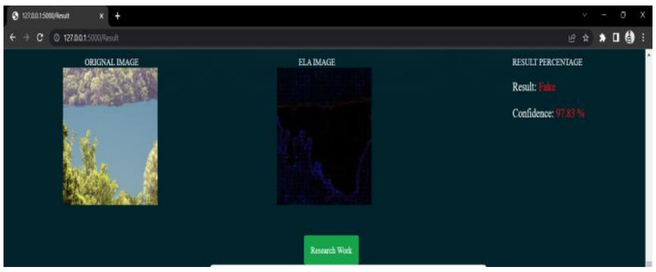


Fig. 10. Prediction of Fake Image

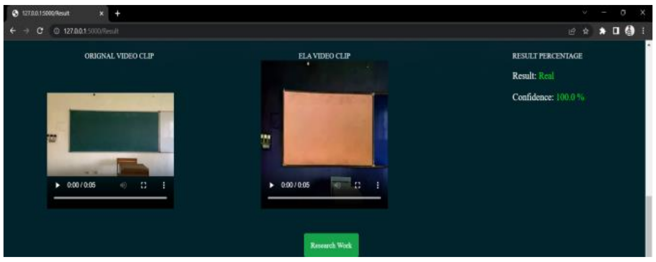


Fig. 11. Prediction of Real Video

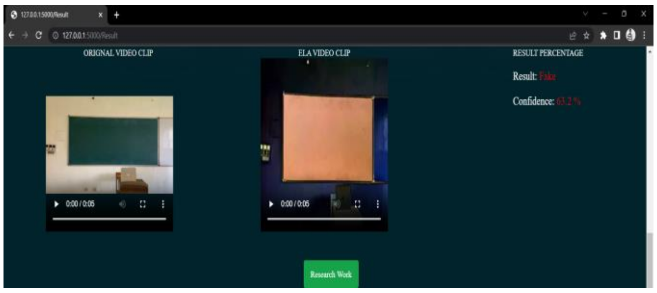


Fig. 12. Prediction of Fake Video

V. CONCLUSION

In summary, machine learning, especially Convolutional Neural Networks (CNNs), is a promising method for detecting manipulated multimedia, including images, videos, and audio, and identifying manipulation types like image tampering, deepfake videos, and audio forgeries. The process involves key steps such as data pre-processing, feature extraction, and model parameter optimization through labeled dataset training. CNNs have showcased their effectiveness in distinguishing between genuine and manipulated content across diverse domains. Nevertheless, challenges persist, such as the need for comprehensive datasets, evolving manipulation techniques, and the potential for adversarial attacks on the models. Notably, we've introduced an innovative video frame detection module with training accuracy at 92.17% and testing accuracy at 87.87%. In audio detection, training accuracy is 97.34%, with testing accuracy at 96.07%, highlighting substantial accuracy enhancements compared to existing systems through unique methods and size adjustments. Future research should explore techniques like transfer learning and ensemble methods, as well as incorporating contextual information.

REFERENCES

[1] David Guera and Edward J. Delp "Deepfake Video Detection Using Recurrent Neural Networks", Video and Image Processing Laboratory (VIPER), Purdue University, 14 February 2019.

[2] Francesco Marra, Diego Gragnaniello, LusiaVerdoliva "A Full-Image Full-Resolution End-to-End-Trainable CNN Framework for Image Forgery Detection", 2019.

[3] Sara Ferreira, Mário Antunes and Manuel E. Correia "Exposing Manipulated Photos and Videos in Digital Forensics Analysis", 2021.

[4] Anh-thu Phan-ho and Florent Retraint "A Comparative Study of Bayesian and Dempster-Shafer Fusion on Image Forgery Detection", 14 September 2022.

[5] Asad Malik, Minoru Kuribayashi, Sani M. Abdullahi, and Ahmad Neyaz Khan "DeepFake Detection for Human Face Images and Videos", 11 February 2022.

[6] S.AnithaJebamani , S.Gomathi, Dr.SomaPrathibha, Deepika Sree D, Preetha S "Detection Of Fake Audio",lkogretim Online - Elementary Education Online,Vol 19 (Issue 4), 2020.

[7] Ameer Hamza, Abdul Rehman Javed, Farkhund Iqbal, Natalia Kryvinska, Ahmad S. Almadhor, Zunera Jalil, and RoubaBorghol "Deepfake Audio Detection via MFCC Features Using Machine Learning", 29 December 2022.

[8] Peng Zhou1 Xintong Han, Vlad I. Morariu,Larry S. Davis "Learning Rich Features for Image Manipulation Detection", 2018.

[9] Kaggle.com, "ASVspoof 2019 Dataset,"[Online], Available: https://www.kaggle.com/datasets/awsaf49/asvpoof-2019-dataset.

[10] Kaggle.com, "CASIA 2.0 Dataset,"[Online], Available: https://www.kaggle.com/datasets/divg07/casia-20-image-tampering-detection-dataset.