International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 9
***Article Received:** 05 July 2023 **Revised:** 25 August 2023 **Accepted:** 15 September 2023*

Scrutinization of Video posts on Social Media for Authenticity

Neelakantam Pavani[1], K. Shyamala[2]
[1]Research Scholar, Department of CSE
Hyderabad, India
mee_pav@yahoo.com
[2]Professor, Department of CSE
Hyderabad, India
prkshyamala@gmail.com

Abstract—Multimedia posts, especially video posts are easy to understand, attractive and are considered as proof of evidence for an event that occurred. Videos are one of the frequently shared posts on social media. Videos and images speak louder than words. They are very much believed by people and forwarded. Videos reach every person despite of language or literacy barriers. Technological improvements lead to easy generation of fraudulent videos to create a negative impact on people and society. Celebrities and politicians are highly affected because of fake video generation and dissemination. The video posts uploaded and forwarded on social media should be analyzed and identified to stop propagation before they create harm to the society. The proposed work converts the video clips into a sequence of frames. The keyframes are then identified by the use of histogram comparison. A CNN model is built with optimum layers for image classification. The keyframes or images are then classified using the CNN model to identify fraudulent content. The proposed work is light in terms of processing when compared to existing or conventional video classification. The work uses FaceForensics++, DeepFake and Celeb-DF V2 datasets and achieved 99.7%, 99.8% and 98.01% accuracy to identify fraudulent video posts.

Keywords-CNN, histograms, fraudulent video, keyframes, OpenCV, social media.

I. INTRODUCTION

The usage of social media is increasing with the increasing use of mobile phones and increasing connectivity all over the world. Social media plays one of the major roles in connecting people across the world. As of January 2022, the leading audience of Facebook is India with about 33 crores of users, United States ranked next with 17.9 crores of users [1]. As per Statista April 2022, 38.5% of the global audience of Twitter users are aged between 25 and 34 years, followed by the next largest age group between 35 and 49 years old, and users below 24 years are almost 21% [2]. [3] state that the worldwide social media users increased from 4.62 billions in Jan 2022 to 4.72 billions in Jan 2023.

The increasing use of social media leads to users posting all kinds of data like text posts, image posts, audio posts and video posts. Rather than text or audio posts; image and video posts are much forwarded as they convey information that is easy to understand and believe. Fraudulent videos sometimes lack normal eye blinking, show some illogical head movements, confusing head postures and some patches on the tampered regions. However, they are difficult to be identified by a normal person.

For example, a social media influencer has been arrested for faking his suicide [4]. Such videos can be created easily these days through applications and tools available on mobile phones. But the impact such videos create on society and people will be very harmful. So, the phoney video posts should be identified and stopped before disseminating through various social media sites.

A lot of research is done on the identification of fraudulent image posts on social media, but less research is done to identify fraudulent video posts. This paper proposes to build a simple system to detect fake video posts on social media. The proposed work converts the videos into a sequence of frames. The keyframes are then selected from the frames. These keyframes or images are then processed like images to identify fraudulent content.

This paper is organized as follows: Section II presents the existing research work to detect fraudulent video posts on social media, and Section III describes the methodologies used and the architecture in the proposed work. Section IV presents the results obtained by applying the developed framework to different datasets. A comparison of the proposed work with existing models is also presented in this section. Section V presents the conclusion and future scope.

II. RELATED WORK

This section presents some of the literature works that were published to discern between real and fake videos on social media. **Korshunov, P et. al. [5]** considered mouth movements of frontal faces to identify fake videos. The authors detected audio-visual inconsistencies and worked on audio manipulations in the speech of the person. Mel-frequency

Cepstral Coefficient (MFCC) is used to extract audio features. They experimented on various classifiers namely MultiLayer Perception (MLP), Gaussian Mixture Model (GMM), Support Vector Machine (SVM) and Long Short-Term Memory (LSTM) and the results obtained were compared. The datasets used were VidTIMIT -10, AMI-977, and GRID -1000 videos of 3 seconds. The work achieved a good accuracy of 99% with LSTM.

Bismi Fathima Nasar et. al. [6] proposed a detection model for fake images, video, and audio. The paper first pre-processes the content and then uses deep learning to classify the posts. The framework is composed of four phases: Data Pre-processing phase, Image Enrichment phase, Development of CNN Model, and Testing Phase. Matplotlib is used to convert audio posts into spectrogram images which are then enhanced using Librosa, OpenCV is used to convert video posts into sequence of frames. Finally, the frames are trained and tested using CNN. The datasets VidTIMIT, DeepfakeTIMIT and Face Forensics++ were used and achieved an accuracy of 99%, 85% and 90% respectively. **Li, Y., Chang et. al., [7],** Irrational Eye blinking in fraudulent videos was focused in this paper. Long-term Recurrent Convolutional Neural Networks (LRCN) was used. The previously learned knowledge was used to make a distinction between open and closed eyes states. The extracted features are fed to a Recursive Neural Network (RNN) with Long Short Term Memory (LSTM) for Sequence Learning. CEW Dataset was used and an accuracy of 99% was achieved.

Sabir, E et. al. [8] proposed a model for detecting fake videos using DenseNet architecture on FaceForensics++ dataset. Three manipulation methods: Deepfake, Face2Face and FaceSwap were tested. The framework consisted of two steps, first: cropping the faces from video frames (explicit and implicit alignment using Spatial Transformer Network (STN)), and the second step was to detect manipulation in facial region. Recurrent Convolutional Network was used for manipulation detection. An accuracy of 96.9%, 94.5% and 96.3% was achieved for three manipulations respectively. **Guera, D. et. al.[9]** proposed a method to detect deepfake videos. The authors use a Convolutional Neural Network (CNN) to extract features of frames. A Recurrent Neural Network (RNN) is trained using the features and learns to classify the real and manipulated videos. For feature extraction CNN is used and LSTM for sequence processing. The dataset used consists of 300 real videos selected from HOHA (Hollywood Human Actions) dataset and 300 fake videos from websites. The work achieved 97.1% accuracy.

Suratkar et. al. [10] the authors performed experiments on pre-trained architectures like ResNetV5, VGG16, EfficientNet, Inception, and Efficient Net with LSTM. The datasets DFDC and Face Forensics ++ were trained and tested. The model is tested and analyzed with residual image input and new test dataset. The efficiency of transfer learning is examined by conducting experiments on simple model and model with transfer learning. The proposed work achieved 99.2% accuracy through EfficientNet on FaceForensics++ dataset.

Umur Aybars Ciftci et. al. [11] introduced FakeCatcher that worked with biological signals from face regions on the nose (middle part), left cheek and right cheek. They used two classifiers SVM and CNN. They experimented with four datasets namely Face Forensics, CelebDF , FaceForensics++, and DeepFake Dataset and achieved 96%, 91.50%, 94.65% and 91.07% accuracies respectively with the CNN classifier. **Alakananda Mitra et. al. [19]** considered three pre-trained models: XceptionNet, Inception V3 and Resnet50. After experiments, XceptionNet was chosen to train and test the FaceForensics++ and DFDC Datasets. They extracted key frames from the videos, detected faces from the keyframes, cropped and resized the faces to 299x299 size and trained them with XceptionNet. With FaceForensics++ dataset, the authors achieved an accuracy of 98.5% and with a custom dataset of FaceForensics++ and DFDC an accuracy of 92.33% was achieved. Puppet-master refer to manipulation of a new video (puppet) as per the facial expressions, head and eye movements of the original video[20]. [21] Divided videos into three types: Portrait video, Clickbait video and Misleading video.

III. PROPOSED WORK

The proposed work aims to achieve the best possible accuracy to identify fraudulent video posts on social media. Once the fraudulent video posts are detected, the proliferation of such posts can be controlled and stopped before it creates any damage to public opinion and further the society. The proposed work is implemented in different stages:

A. Converting Video into a Sequence of frames and changing colour space

The first step in video classification is to convert the video into a sequence of frames. Python provides an OpenCV library for image processing and video processing. First, a capture object is created, which captures the video to be processed and helps in performing various operations on videos. The VideoCapture() method of the cv2 module of the OpenCV library is used to create the capture object. The read() method of the VideoCapture object is called iteratively to read all the frames of the video. The read function returns two values, one is ‘ret’ which is a Boolean value, which is true if the frame is present or it returns false when the frames are all identified. The second parameter is the frame as an image. The waitKey() method of the cv2 module can be used to restrict the capture of frames from the video. The set() method of capture object helps to change the number of frames read per second, thus reducing the number of frames read. This is required as the number of frames increases, the processing load also increases. The proposed work uses waitKey() to reduce the number of frames extracted.

***Article Received:** 05 July 2023 **Revised:** 25 August 2023 **Accepted:** 15 September 2023*

To identify the key frames from the frames extracted, we need to compare the adjacent frames. The read() method is called twice to capture two consecutive frames. They are saved as prev_frame and curr_frame, to calculate the similarity between the consecutive frames. The cvtColor() method of OpenCV is used to convert the colour space of an image. The colour space of the consecutive images is converted from RGB or BGR to HSV (Hue Saturation Value). OpenCV provides above 150 colour space conversions. The HSV colour space closely corresponds to the human visual perception of colour and is ideal for processing videos based on colour descriptions. Hue is the angle between 0-180 degrees around the red axis which determines the actual colour, Saturation is the depth of colour and is measured from the central axis to the outer surface (0-100%). Value is the brightness of the colour, it also measures from 0-100%. The HSV colour space is ideal for image processing and thus video processing.

B. Calculate histograms of the frames and normalize.

Since each video produces a large number of frames, it is infeasible to process all the frames of the video. Only selected key-frames that are unique and exhibit maximum information unlike the similar frames should be identified and extracted. Frames with large variations in a video are selected as keyframes. Hence, selected key-frames of the video are used to classify real or fake video. Keyframes can be identified from the set of frames using histograms. A Histogram represents the pixel intensities in an image (colour or grayscale). A histogram represents a vector whose components represent similar colours in an image.

The calcHist() function of the cv2 module is used to calculate histograms of the images. A histogram represents the frequency of pixels in an image. The calcHist() method takes five parameters, the image, channels (HSV) of the image, the mask to calculate the histogram of only specific portions of the image, the histsize that contains the number and size of bins specified as a list, the last parameter of calcHist() specifies the range of possible pixel values, it is different for different color spaces like RGB, HSV etc. The proposed work gives the hsv image as a parameter to calcHist, masking is not specified in the proposed work, and the full frame is considered. The histograms are then normalized, to increase the image contrast.

The normalize() method of OpenCV is used to normalize the histograms. The method accepts seven parameters, they are: src, dst, alpha, beta, norm_type, dtype and mask. The first two parameters are input and output images. alpha specifies the lower and beta specifies the upper value for the normalization range, norm_type is the type of normalization, dtype is the data type of output and mask is used when only a part of the image is required for processing, this is optional. The proposed work uses the NORM_MINMAX type that normalizes image pixels to a range [0,1]. The histograms are normalized with alpha=0 and beta=1.

C. Compare histograms to select keyframes

The compareHist() of cv2 is used to compare the normalized histograms. This method takes 3 arguments: the previous frame, the current frame and the flag that indicates the method used for comparison. OpenCV provides several built-in methods to compare histograms

- **HISTCMP_CORREL**: this method is used to calculate the correlation between two histograms.
- **HISTCMP _INTERSECT**: this method computes the intersection between the histograms.
- **HISTCMP _CHISQR**: this method finds the Chi-Squared distance between the two histograms.
- **HISTCMP _CHISQR_ALT**: computes the alternative Chi-Square for histograms
- **HISTCMP _HELLINGER:** method is a Synonym for **CV_COMP_BHATTACHARYYA**
- **HISTCMP _KL_DIV:** calculates the Kullback-Leibler divergence
- **HISTCMP _BHATTACHARYYA**: this method computes the “overlap” between the histograms.

The compareHist() method takes histograms of consecutive frames as input and returns a metric value. The metric value returned by the compareHist() method with the **Correlation** or **Intersection** flag specified is higher, when the histograms of images are more similar. The metric value returned with **chi-square** or **Bhattacharyya** specified as flag, indicates higher similarity for lower metric value. Experiments proved that, for the datasets used in the proposed work, HISTCMP_INTERSECT gave better extraction of keyframes. Since INTERSECT gave a higher metric value for higher similarity, the frames whose comparison metric value is lesser than a selected threshold are selected as the keyframes. Thus, fewer frames with variations were selected as the keyframes. The threshold value is different for different datasets. The threshold is selected by retrieving and analyzing the metric values of all histogram comparisons.

Figure 1 is a picture of one of the selected keyframes from a fake video of the Celeb-DF version 2 dataset [22] and Figure 2 is the result of converting the same keyframe from RGB to HSV color space. cvtColor() method is used for this conversion. The proposed method gave better results working with HSV images than RGB images. Hence, all the keyframes are converted to HSV colour space. The next step is to calculate histograms of all the keyframes using the function calcHist(). The histograms present the pixel intensities of the colours in the image. The x-axis depicts the colour space and the y-axis depicts the frequency of pixels. Since the frequency of pixels varies from image to image; for comparison, the histograms should be made uniform by normalization.

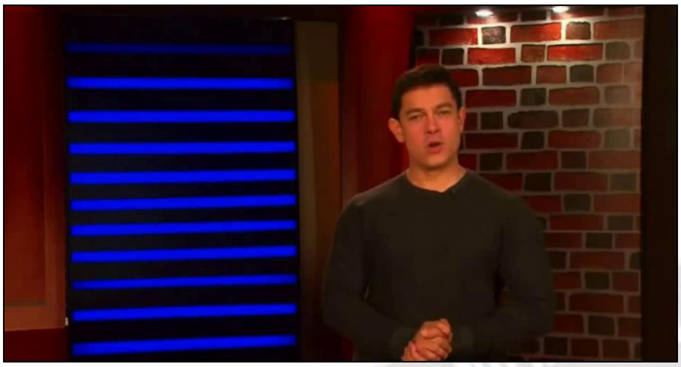


Figure 1. Fake Key Frame from Celeb-DF-V2 Dataset

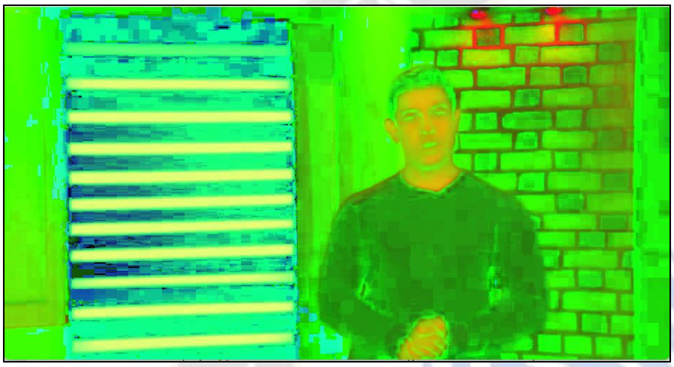


Figure 2. Frame in Figure 1 converted to HSV Color Space

Figure 3 presents the image histogram of Figure 2 and its normalized histogram which is normalized between 0 and 1.0. Since the image is brighter, both the histograms show peaks in the higher part of the x-axis. The normalize() method with NORM_MINMAX type is used to normalize the frames.

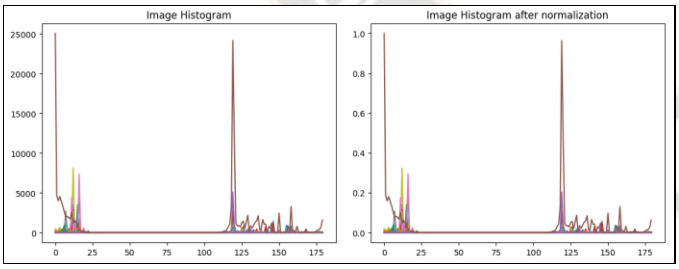


Figure 3. Histogram and Normalized histogram of the frame in Figure 2

Figure 4 is a picture of one of the selected keyframes from a real video of the Celeb-DF version 2 dataset. Figure 5 is the same image converted to HSV colour space. Figure 6 represents the histogram of the image in Figure 5 and its normalized histogram. It can be noted that; since this is a darker image, there are more peaks in the lower part of the x-axis. Then, the normalized histograms are compared using the compareHist() method with intersect as the comparison. This gives a higher metric for more similar frames. Hence through analysis of all metric values, a threshold of 30.0 is decided and all the keyframes that result in a metric value lower than the threshold are selected as keyframes.

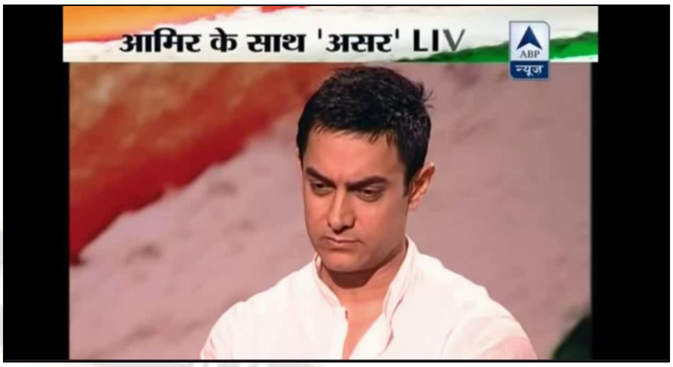


Figure 4. Real Key Frame from Celeb-DF-V2 Dataset

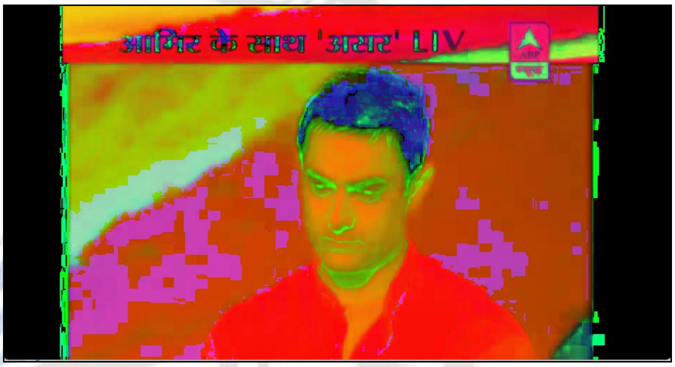


Figure 5. Frame in figure 4 converted to HSV Color Space

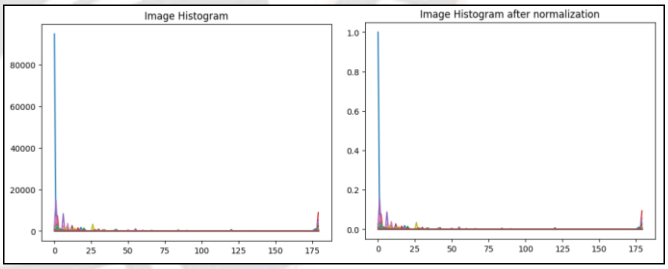


Figure 6. Histogram and Normalized histogram of the frame in Figure 5

D. Calculate Error Level Analysis of the Images (Keyframes)

The selected Key Frames are then processed like images. The keyframes are converted to ELA [16][17] images as they give better results for CNN classification than normal images. Error Level Analysis (ELA) of images helps to identify the tampered portions of the image by bloating or highlighting the tampered part in the image. This is possible because of the different compression levels of tampered images. For an original image, all portions of the image will have the same compression levels, but when an image is tampered with and saved multiple times, the compression levels of the tampered

Article Received: *05 July 2023* ***Revised:*** *25 August 2023* ***Accepted:*** *15 September 2023*

part and the non-tampered part vary; this is identified and projected in ELA images. Thus the selected key frames of all videos for training and validation are converted to ELA images.

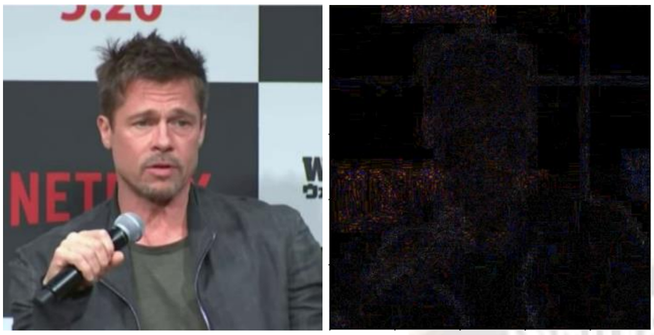
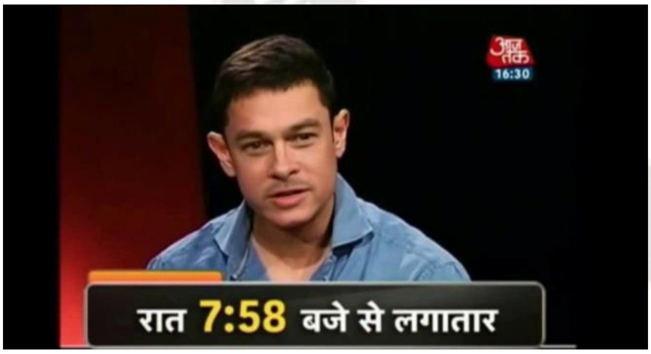


Figure 7. Real Key-frame from CelebDF dataset and its ELA image

Figure 7 shows a selected keyframe from a real video and its ELA image. It can be noticed that the Error Level Analysis of the original key frame is dark and does not highlight any part of the image. Figure 8, is a selected keyframe from Fake video clip of Celeb DF V2 dataset and its ELA image. The ELA image shows the error pattern that highlights the tampered part of the image. It is noticed that the facial (eyes, nose and mouth) region of the face in the ELA is shown bright and bloated. Thus, ELA images prove to be better input to CNN models for real and fake classification. ELA highlights the differences in compression rates of an image. The differences are represented as high-contrast edges. ELA increases the accuracy of the CNN model to a great extent and is thus best suited combination with CNN model for detecting fake content in images.



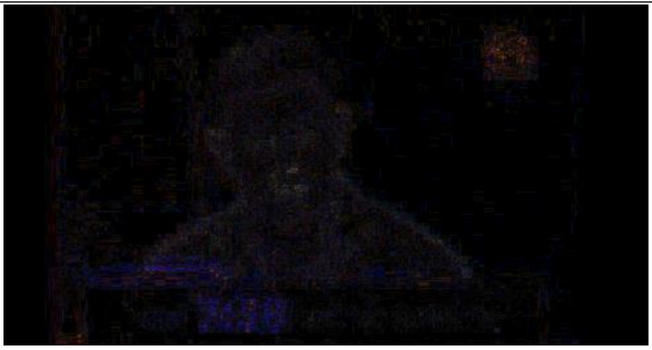


Figure 8. Fake Key-frame from CelebDF dataset and its ELA image

Finally, the key frames of the input videos are given as input to the CNN model for training and validation purposes. Figure 9 describes the step-by-step procedure to pre-process and classify real and fake videos.

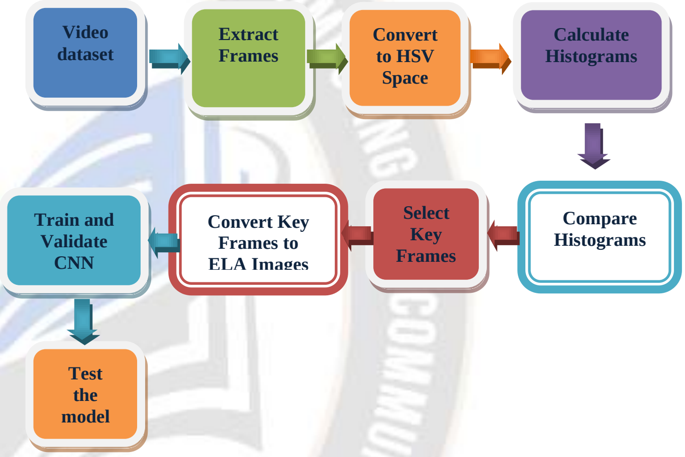


Figure 9. Proposed model for video classification

E. Developing and training the model

After the keyframes are identified and pre-processed using Error Level Analysis like images, the next step is to build the CNN model. Various experiments were conducted for tuning the hyperparameters like learning rate, epochs, dropouts etc. The best-performing layers were chosen from literature survey and by experimental analysis. Finally, the best combination of layers is selected for the CNN that gives an optimum accuracy of the keyframes. Figure 10 is the final CNN model used for the classification of keyframes.

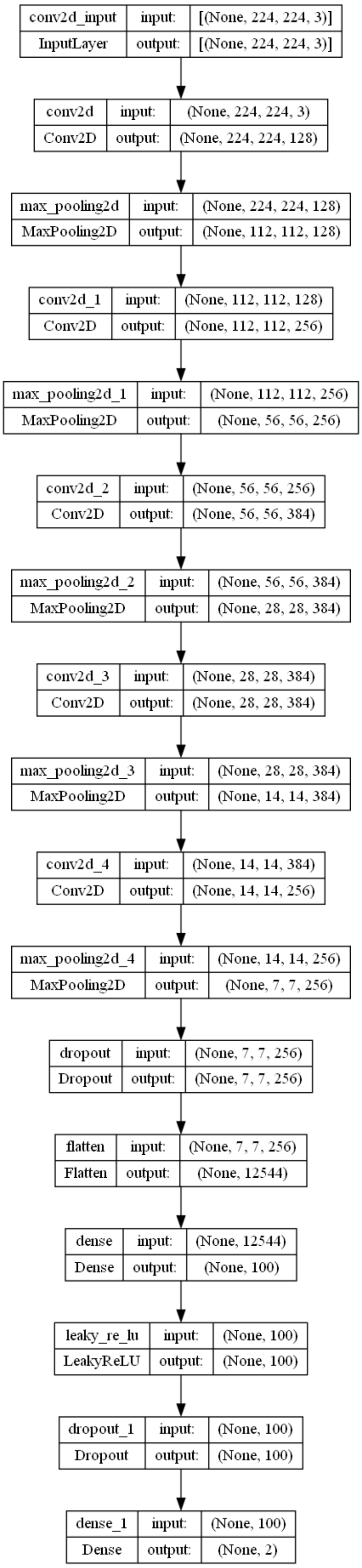

conv2d_input | input: | [(None, 224, 224, 3)]
InputLayer | output: | [(None, 224, 224, 3)]
conv2d | input: | (None, 224, 224, 3)
Conv2D | output: | (None, 224, 224, 128)
max_pooling2d | input: | (None, 224, 224, 128)
MaxPooling2D | output: | (None, 112, 112, 128)
conv2d_1 | input: | (None, 112, 112, 128)
Conv2D | output: | (None, 112, 112, 256)
max_pooling2d_1 | input: | (None, 112, 112, 256)
MaxPooling2D | output: | (None, 56, 56, 256)
conv2d_2 | input: | (None, 56, 56, 256)
Conv2D | output: | (None, 56, 56, 384)
max_pooling2d_2 | input: | (None, 56, 56, 384)
MaxPooling2D | output: | (None, 28, 28, 384)
conv2d_3 | input: | (None, 28, 28, 384)
Conv2D | output: | (None, 28, 28, 384)
max_pooling2d_3 | input: | (None, 28, 28, 384)
MaxPooling2D | output: | (None, 14, 14, 384)
conv2d_4 | input: | (None, 14, 14, 384)
Conv2D | output: | (None, 14, 14, 256)
max_pooling2d_4 | input: | (None, 14, 14, 256)
MaxPooling2D | output: | (None, 7, 7, 256)
dropout | input: | (None, 7, 7, 256)
Dropout | output: | (None, 7, 7, 256)
flatten | input: | (None, 7, 7, 256)
Flatten | output: | (None, 12544)
dense | input: | (None, 12544)
Dense | output: | (None, 100)
leaky_re_lu | input: | (None, 100)
LeakyReLU | output: | (None, 100)
dropout_1 | input: | (None, 100)
Dropout | output: | (None, 100)
dense_1 | input: | (None, 100)
Dense | output: | (None, 2)

Figure 10. CNN model to classify the keyframes as real or fake

The model consists of Convolutional layers, MaxPool2D layers, Dropout and fully connected layers. The activation functions used are the ‘swish’ in the inner layers and ‘softmax’ in the last dense layer. The swish activation function is a smooth, non-monotonic function that gives consistent results, better than ReLU [14].

Figure 10 presents the pictorial representation of the CNN model developed. This figure is generated using plot_model(). This representation displays the layer names. Figure 11 represents the visualization of the CNN model using the layered_view() method of visualkeras. This is another representation to visualize the CNN model.

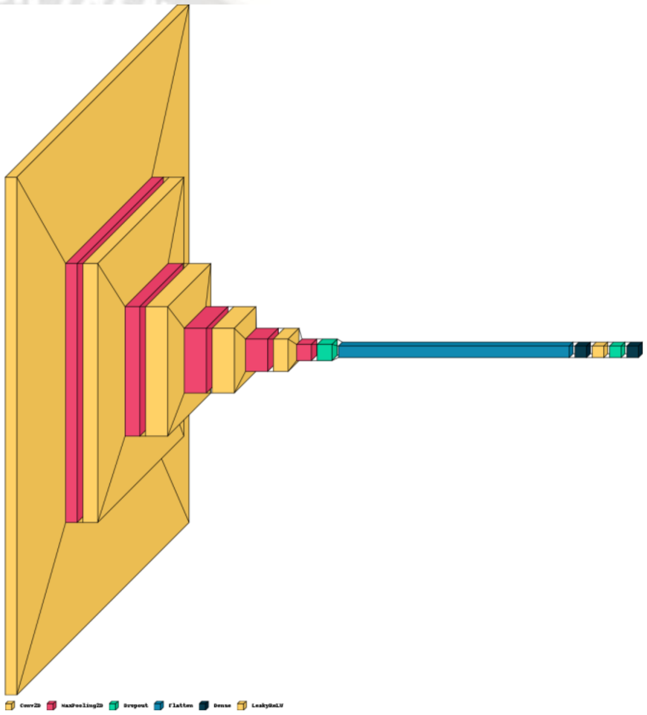


Figure 11. Visualization of the CNN model using VisualKeras

CNN or the Convolution Neural Network [18] is a Deep Learning method that is used to classify the real and fraudulent keyframes. The CNN model takes the real and fraudulent keyframes as input and learns the features of both the categories of keyframes. Then a different set of videos are tested and classified by CNN in the same manner. CNN consists of Convolution Layer, MaxPooling and Fully Connected Layer.

Feature Extraction of keyframes is done by convolution layer which gives feature-maps as output. The dominant features are extracted by reducing dimentionality using Pooling layer. Max Pooling, Min Pooling and Average Pooling are three ways of reducing the dimensions. Dense layer or Fully-connected layer connects every neuron in one layer to every other neuron in another layer. The Dense layer gives the best predictions for

***Article Received:** 05 July 2023 **Revised:** 25 August 2023 **Accepted:** 15 September 2023*

correct classification. The output of CNN model is a vector of probabilities that present the final classification accuracy of the model on the given test dataset.

The proposed work uses three datasets: Deepfake dataset, Celeb-DF V2 [15][22] and FaceForensics++. 15 real video clips and 15 fake video clips are taken to train, validate the model and three unseen videos to test the model. FaceForensics++ is identified as the highest used dataset for video classification till 2020 [12]. The video clips of DeepFake range from 3-4 seconds each. Each of the video clips of the FaceForensics++ dataset ranges from 6-34 seconds.

Once the model is trained and validated, the next step is to test the dataset. A different set of videos are given for testing the model. Since, the model processes images, when multiple videos are given as input to test the model, the first step is to extract the frames and select the keyframes for all test videos. Then the key frames belonging to each video should be grouped separately as the result of the all key frames of a video is the result of the video classification. Hence, the keyframes belonging to one video are identified using the fnmatch() function.

The fnmatch() function is used to match the filenames for pattern matching. A separate folder is created for each video file and all the identified keyframes are moved into the folder using shutil.move() method. Then the key frames belonging to a video are sent to the model for testing and prediction. The fusion of probabilities of all the key frames of the video is taken as the accuracy of the classification of that video.

IV. RESULTS AND ANALYSIS

The test dataset consists of 3 real videos and 3 fake videos. The approach extracts around 100 keyframes for each video. The model is trained with all the selected keyframes. Finally, a different set of videos is given for testing the model. The keyframes for the test set are also extracted and finally, the fusion of probabilities of the keyframes is taken as the result of the video classification.

Video processing takes a lot of computational power and memory. Giving the video dataset as input to the CNN model to classify real and fake is a very complex task. It takes a lot of computational power and is infeasible to perform all videos posted on social media. Thus, this work brings up a very simple approach where the video clip is first divided into frames, Thousands of frames are extracted from a 3-4 seconds video, and then the key frames are identified from the extracted frames using histogram comparison. The keyframes are given as input to CNN for training and validation of the model. This approach is very simple and reduces the processing load. With computational efficiency as a tradeoff, this model is best suited to identify fake videos. The model is trained, validated and tested on four video datasets: the Face Forensics++ dataset with Face2Face manipulation consisting of 1000 original video clips and 1000 manipulated video clips, the Face Forensics++ dataset with FaceSwap manipulation consisting of 1000 original and 1000 manipulated video clips, third is the Deepfake TIMIT video dataset consisting of 800 fake video clips and finally the Celeb-DF version2 dataset with 590 original video clips and 5639 fake video clips. The work is trained and tested with only 15 original and 15 fake videos from each dataset. The model is then tested with 3 different test videos and obtained the results as shown in Table 1. All the video clips range in size from 3 seconds to 50 seconds.

TABLE I. COMPARISON OF THE PROPOSED MODEL AGAINST FOUR VIDEO DATASETS

Dataset	**Gen. Model**	**Configuration**	**Real Accuracy %**	**Fake Accuracy %**
Face Forensics++	Face2Face	Real-14 videos, Fake- 15 videos	99.7	99.5
Deepfake TIMIT	Default	Real-3 videos, Fake- 15	99.8	99.2
Face Forensics++	Faceswap	Real-20, Fake- 20	99.1	99.4
Celeb-DF-V2	Default	Real-15 Fake-15	96.02	100

Table 1 presents the results obtained by the proposed models on testing the four different datasets. The model gave good results on all the datasets. Figure 12 presents the training accuracy and validation accuracy graphs of the real and forged videos in the test set of Celeb DF V2 dataset.

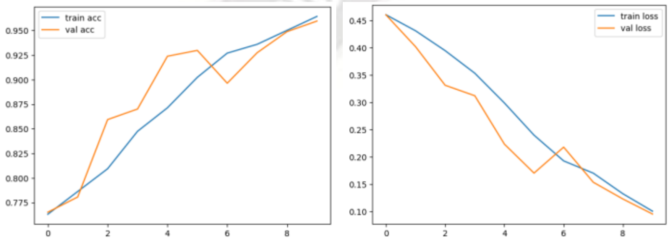


Figure 12. The training and loss accuracy graphs of real and forged videos of the Celeb DF V2 dataset.

TABLE II. COMPARISON OF THE PROPOSED MODEL AGAINST EXISTING MODELS

Paper	**Dataset**	**Data set Size**	**Methodol ogy**	**Epochs**	**Accura cy**
Bismi Fatima [6]	VidTIMIT , Deepfake TIMIT, FaceForen	1000 video clips	Keyframe extraction , CNN	15	99%, 85%, 90%

	sics++				
Ekraam Sabir [8]	Face Forensics-Deepfake, Face2Face, FaceSwap	1000 video clips	Face detection, cropping and alignment, CNN, RNN	Not Mentioned	96.9, 94.35, 96.3
Darius Afchar [13]	DeepFake, Face2Face	300 video clips	MesoNet	Not mentioned	98.4%, 95.3%
Proposed model	FaceForensics++ - Face2Face	15 Real and 15 Fake video clips	Key frame extraction, CNN	15	99.6%

Table 2 compares the test results of the proposed work with existing state of art models. The table presents the methodologies used, datasets used, dataset size, number of epochs and the accuracy obtained by all models.

The result obtained by the proposed model is better than existing models. The proposed model is built and works with very little computational power and gives better comparative results in identifying fake videos.

V. CONCLUSION

The proposed model achieved optimum results at low computational costs. It performed efficiently to classify real and fake audio with a minimum number of epochs. The model converts the videos into keyframes and then gives the keyframes as input to an optimum CNN model for classification. The model achieved 98.01% accuracy in classifying real and fraudulent videos of the Celeb DF V2 dataset, 99.25% with FaceForensics++ dataset with FaceSwap manipulation, 99.6% for FaceForensics++ dataset with Face2Face manipulation and 99.5 for Deepfake TIMIT dataset.

REFERENCES

[1] https://www.statista.com/statistics/264810/number-of-monthly-active-facebook-users-worldwide/. last accessed 2023/08/06.

[2] https://www.statista.com/statistics/376128/facebook-global-user-age-distribution/. last accessed 2023/08/06.

[3] https://www.smartinsights.com/social-media-marketing/social-media-strategy/new-global-social-media-research/#:~:text=A%20summary%20of%20global%20social,of%20%2B137%20million%20users%20YOY. last accessed 2023/08/06.

[4] https://timesofindia.indiatimes.com/videos/city/mumbai/mumbai-social-media-influencer-arrested-for-faking-his-suicide/videoshow/84723993.cms, last accessed 2023/08/06.

[5] Pavel Korshunov, F., Sébastien Marcel, S.: Speaker Inconsistency Detection in Tampered Video. 26th European Signal Processing Conference (EUSIPCO), (2018).

[6] Bismi Fathima Nasar, F., Sajini T, S., Elizabeth Rose Lason, T.: Deepfake Detection in Media Files - Audios, Images and Videos. IEEE Recent Advances in Intelligent Computational Systems (RAICS) | December 03-05 (2020).

[7] Yuezun Li, F., Ming-Ching Chang, S., Siwei Lyu, T.: In Ictu Oculi: Exposing AI Generated Fake Face Videos by Detecting Eye Blinking. IEEE International workshop on information forensics and security (WIFS). IEEE (2018).

[8] Ekraam Sabir, F., Jiaxin Cheng, S., Ayush Jaiswal, T., Wael AbdAlmageed, Iacopo Masi, Prem Natarajan.: Recurrent Convolutional Strategies for Face Manipulation Detection in Videos. Interfaces (GUI), 3(1), 80-87. (2019).

[9] Guera, D., & Delp, E. J. (2018). Deepfake Video Detection Using Recurrent Neural Networks. 2018 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS). doi:10.1109/avss.2018.8639163

[10] Suratkar, Shraddha, and Faruk Kazi. "Deep Fake Video Detection Using Transfer Learning Approach." Arabian Journal for Science and Engineering (2022): 1-11.

[11] Umur Aybars Ciftci, ˙Ilke Demir, and Lijun Yin.: FakeCatcher: Detection of Synthetic Portrait Videos using Biological Signals. IEEE transactions on pattern analysis and machine intelligence, vol. X, no. X, july 2020.

[12] MD SHOHEL RANA, MOHAMMAD NUR NOBI, BEDDHU MURALI and ANDREW H. SUNG. “Deepfake Detection: A Systematic Literature Review”. IEEE Access March 10, 2022.

[13] Darius Afchar, Vincent Nozick, Junichi Yamagishi, Isao Echizen. “MesoNet: a Compact Facial Video Forgery Detection Network”. HAL Id: hal-01867298 https://hal-upec-upem.archives-ouvertes.fr/hal-01867298 4 Sep 2018

[14] Bhuvanesh Singh, F., Dilip Kumar Sharma, S.: Predicting image credibility in fake news over social media using multi-modal approach. Neural Computing and Applications. Springer Nature (2021).

[15] Yuezun Li, Xin Yang, Pu Sun, Honggang Qi and Siwei Lyu .Celeb-DF (v2): A New Dataset for DeepFake Forensics**.** IEEE Conference on Computer Vision and Patten Recognition (CVPR), Seattle, WA, United States, 2020.

[16] Muhammed Afsal Villan, F., Kuncheria Kuruvilla, S., Johns Paul, T., Prof. Eldo P Elias.: Fake Image Detection Using Machine Learning. IRACST - International Journal of Computer Science and Information Technology & Security (IJCSITS), Vol.7, No. 2, (2017).

[17] Bhuvanesh Singh, F., Dilip Kumar Sharma, S.: Predicting image credibility in fake news over social media using multi-modal approach. Neural Computing and Applications. Springer Nature (2021).

[18] KHALID M. HOSNY, AKRAM M. MORTDA, MOSTAFA M. FOUDA and NABIL A. LASHIN. “An Efficient CNN Model to Detect Copy-Move Image Forgery”. IEEE Access May 2022.

[19] Mitra, Alakananda, Saraju P. Mohanty, Peter Corcoran, and Elias Kougianos. "A machine learning based approach for

deepfake detection in social media through key video frame extraction." SN Computer Science 2 (2021): 1-18.

[20] Agarwal, Shruti, Hany Farid, Yuming Gu, Mingming He, Koki Nagano, and Hao Li. "Protecting World Leaders Against Deep Fakes." In CVPR workshops, vol. 1, p. 38. 2019.

[21] Li, Xiaojun, Shaochen Li, Jia Li, Junping Yao, and Xvhao Xiao. "Detection of fake-video uploaders on social media using Naive Bayesian model with social cues." Scientific Reports 11, no. 1 (2021): 16068.

[22] Karandikar, Aarti, Vedita Deshpande, Sanjana Singh, Sayali Nagbhidkar, and Saurabh Agrawal. "Deepfake video detection using convolutional neural network." International Journal of Advanced Trends in Computer Science and Engineering 9, no. 2 (2020): 1311-1315.