Project Report For:

AI Generated Video Detection using CNN & RNN

By -

Atharva Kasar

Abhishek Prabhu

Anjor Rane

Deep Tandel

Abstract

The need to provide trustworthy techniques for differentiating between real and AI-generated videos has arisen from the emergence of AI-generated media. This difference has become more hazy due to generative models like deepfakes and GANs. This project proposes a hybrid AI model that combines convolutional and recurrent neural networks to address this. the CNN will extract spatial data from video frames, while the RNN will examine temporal sequences and identify disparities in AI generated videos. An extensive dataset of real and AI generated videos will be carefully selected from copyright-free sources, labeled, and balanced. Robustness will be ensured by preprocessing approaches such as data augmentation, resizing, normalization, and frame extraction. Using measures like loss and accuracy, the model will be adjusted to maximize feature extraction and sequence learning. We will use a confusion matrix to study misclassifications, concentrating on instances where videos produced by AI and genuine videos have a lot in common. Media verification, content moderation, and digital forensics may benefit from this research .

Keywords:
AI-generated videos, CNN, RNN

Contents

List of Figures

Chapter 1

Introduction

Media production is changing as a result of artificial intelligence (AI), particularly with the emergence of deepfake algorithms, generative models, and Generative Adversarial Networks (GANs). These developments raise serious questions regarding the legitimacy of digital media even while they make it possible to produce incredibly lifelike AI-generated videos. It's become harder to tell the difference between artificial intelligence-generated and real content, which poses major threats like deception, identity theft, and political influence. Therefore, preserving the integrity of digital material depends on having trustworthy and effective algorithms to identify AI-generated videos.

1.1 Motivation

Artificial intelligence (AI)-generated videos are increasingly being misused for nefarious ends, endangering society through cybercrime, fake news, and invasions of privacy. The increasing difficulty in differentiating between real and synthetic media underscores the pressing need for sophisticated detection methods, which is why our study aims to train a model that can accurately recognise AI-generated videos or real videos.

1.2 Objective

As a result, in order to compare the real and artificial intelligence-generated videos, it is necessary, within the framework of the project, to train a model for spatial features extraction using CNNs and temporal sequence analysis using RNNs.. Yet this technology presented in this article aims at reliably detecting fake media in real-time, thus slowing down the circulation of such content.

1.3 Contribution

To address the problem of detection based on videos generated by AI, this research proposes a framework that combines CNN and RNN. RNN's store the motion pattern and temporal relationship between two successive frames while CNN is utilized to learn spatial features from a single frame of a video sequence. The suggested model gives lending information for the difference of real videos and fake videos generated by AI; spatial + temporal. Moreover, it is going to provide more diverse and well-annotated set of both

real and AI generated videos that can be used to fine-tune the model to new types of AI-generated content.

1.4 Problem Definition

It has become more challenging to discern between real and AI generated media due to the quick development of AI-generated media, especially videos produced by deepfake technologies and GANs. The increasing complexity of AI-generated videos, which can accurately mimic real-world features like lighting, motion patterns, and facial expressions, is too much for conventional detection techniques to handle. Given the growing usage of AI-generated videos for nefarious purposes like disinformation, identity theft, and political manipulation, this poses a serious threat to the integrity of digital media. Thus, there is a pressing need to create a reliable, automatic technique for differentiating between actual and artificial intelligence-generated films.

1.5 Organization of Report

Chapter 2 deals with literature survey of Hough Transform and GPU. Chapter 3 contains detailed working of Hough Transform for line and circle detection. In Chapter 4 GPU Computing using MATLAB is explained along with the information about MATLAB Parallel Computing Toolbox and how it can be utilised to access GPU. Chapter 5 explains the design algorithm. The results are discussed in Chapter 6 and Chapter 7 concludes the work.

Chapter 2

Literature Survey

- This research, presented in the paper [1], the research presents AIGVDet, a unique spatiotemporal anomaly detection approach in order to differentiate artificial intelligence-generated videos from real videos, the research presents AIGVDet. It makes use of two ResNet50 sub-detectors: one detects temporal irregularities in optical flow across frames, while the other detects spatial anomalies in frames. Combining these streams improves the model's detection of AI-generated content. Additionally, a brand-new, extensive Generated Video Dataset (GVD) including more than 11,000 videos was produced for testing. Even in compressed videos, the model works effectively, identifying minute imperfections that are invisible to the human eye, such as strange frame transitions.

- This research, presented in the paper [2], suggests a deepfake detection model which integrates CNN, LSTM, and CapsuleNet. Whereas, CapsuleNet preserves face features and motion; CNN extracts spatial features from frames; while LSTM deals with the temporal differences of frames. The model surpassed traditional methods such as XceptionNet with 88 percentage validation accuracy and 95.1 percentage AUC after training the model with the DFDC dataset. Outside of overall performance, additional interpretability was given from Grad-CAM which showed areas in the video that influenced the prediction. Low quality or the rotated deep fakes are picked well by the model and in the future, the model will be enhanced with the help of increased and optimized data set.

- This research, presented in the paper [3], the research analyzes the CIFAKE dataset, which contains 100,000 genuine and false images, to investigate CNN and Vision Transformers (ViT) for AI-generated image detection. With six convolutional layers and three fully linked layers, the best CNN model outperformed ViT, which earned 87.09 percent accuracy because of its low input resolution (32x32). Grad-CAM was utilized to interpret the CNN model, emphasizing that the network concentrated on artificial intelligence (AI)-generated image backdrop artifacts. CNN's depth and carefully calibrated hyperparameters are why it performs better. Future research will investigate novel generative models and higher-resolution datasets in an effort to improve model robustness.

- This research, presented in the paper [4], the study proposes a hybrid model for deepfake detection that combines CNNs and RNNs and is optimized using particle swarm optimization (PSO). RNNs examine temporal connections between frames,

whereas CNNs record spatial anomalies in video frames. For increased accuracy, PSO adjusts hyperparameters such as filter sizes and learning rates. The accuracy of the model was 97.26 percent on the Celeb-DF dataset and 94.2 percent on the DFDC dataset. The model's performance was considerably enhanced via PSO optimization as compared to alternative techniques. According to the report, this hybrid technique can successfully identify deepfakes; further research should concentrate on growing datasets and enhancing real-time detection.

- This research, presented in the paper [5], the study focuses on face detection in video frames and introduces a CNN-based method for identifying deepfake videos. Three steps make up the approach, which uses the DFDC dataset: Faces are extracted from frames using OpenCV and Haar Cascade for preprocessing, features are extracted using ResNet-50 for fine-grained visual characteristics, and a CNN model trained on 3000 actual and 3000 fake frames is used for classification. The model outperformed techniques like random forest and SVM, achieving 98 percent accuracy. The study demonstrates how deep learning methods, such as CNNs and ResNet-50, can enhance deepfake detection and have real-time application potential.

- This research, presented in the paper [6], the research proposes a CNN-RNN hybrid deepfake detection technique. While RNN examines temporal differences across frames to find odd transitions, CNN identifies visual patterns and artifacts by extracting spatial characteristics from video frames. After being trained on the Celeb-DF and DFDC datasets, the model outperformed individual CNN or RNN models, achieving 96.5 percent accuracy on DFDC. It did a fantastic job of picking up on minute variations in face expressions and illumination. Future research will concentrate on real-time performance and investigate additional media manipulation strategies, as the study emphasizes the complementary effects of spatial and temporal analysis in deepfake identification.

- This research, presented in the paper [7], The technique, which uses optical flow-based CNNs to detect deepfake videos, focuses on recognizing temporal anomalies from motion patterns between frames. This method performs better in identifying tiny motion disparities, expanding detection capabilities beyond conventional image-based techniques. Even if it works well, the technique has drawbacks, such as high processing requirements and the possibility of overfitting to particular kinds of deepfakes. The work highlights the necessity of additional motion-based elements in future research. Training datasets should be expanded in future research to improve the detection model's scalability and generalization.

- This research, presented in the paper [8] The study conducted proposes a shallow/temporal anomaly based deepfake dete Whereas CNNs take a single random frame of the video to detect deepfakes, the proposed shallow/temporal anomaly based deep fake detection framework uses CNNs in combination with LSTM RNNs which work on the temporal domain and analyse a sequence of frames of the video. In contrast to the frame-based approach where a single frame is examined in detail, we gain a much higher detection accuracy, thanks to combination of spatial context and temporal dependencies. While it has proven to be efficient, the complexity of the model may become a problem for implementation in real time and adaptability to new deepfake types. This fact is evidenced by experimental outcomes showing

that its capacity to acknowledge AI-generated content is rather dominant. To enhance generalization in future works, larger training sets should be incorporated and computational density should be maximised.

2.1 Outcome of Literature Survey

- The second paper captures the temporal aspect through LSTM and utilize CNN for spatial features capture; by analysing both spatio-temporal aspects together they represent the efficiency of hybrid models which are combination of CNN, LSTM or RNN architectures more easy effective against deepfake detection as evident from literature survey. Techniques like AIGVDet, which mines abnormalities in optical flow and frame shifts, with hybrid models such as CapsuleNet or particle swarm optimisation (PSO) have continued to increase on the accuracy and generalisation over conventional means. Some models focus on optimisation of dataset and hyperparameter whereas some use CNN using Grad-CAM providing explainability through identification of regions which contribute to detection. To enhance detection accuracy and enable the potential for real-time applications, spatiotemporal consistency in spatial domain and temporal aspects through optimized datasets from nonlinear information reduction techniques while enforcing computations with efficient models are promising as illustrated across reviewed research.

Chapter 3

System Design

3.1 Methodology

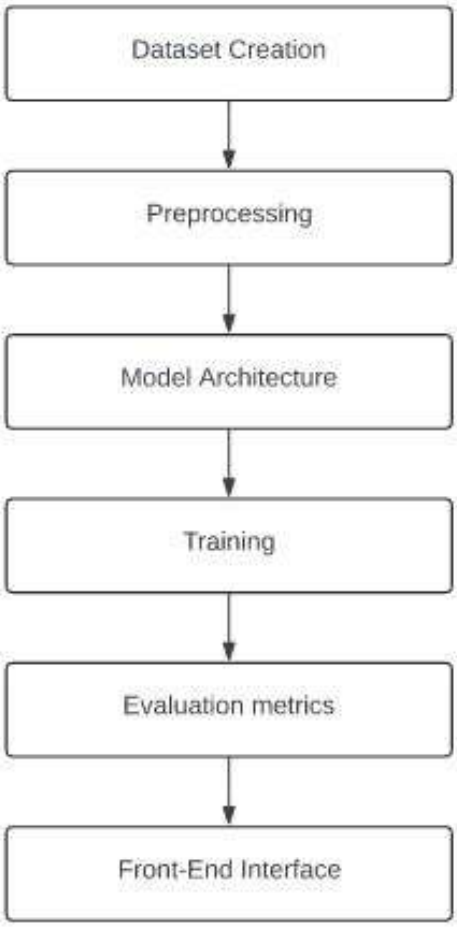

Figure 3.1: Methodology

In this section, the Authors propose the methodology that was employed in order to differentiate the videos created by AI from the real ones based on the CNN/RNN hybrid

model. The proposed methodology involves several stages: namely data collection, data preprocessing, model design, model training, model evaluation.

1. Dataset Creation

   The first is the creation of a dataset which consist real and AI-generated videos. This dataset shall be used for training the model to differentiate between real and artificial intelligence-generated videos. The dataset shall be balanced, having an equal numbers of real and AI-generated videos. Real videos are retrieved from copyright-free websites to maintain data authenticity and to ensure that the data is not biased towards certain sources nor bound by any legal constraints. Similarly, AI-generated videos will be obtained from web archives that allow access to the generated media without copyright. Then, the frames for the video will be extracted so that deep learning models may have easier access to and analyze them.

   It will be possible to use a Python script together with the OS library to automate the data labelling process. This script scans directories with both real and AI videos and sorts them by folders with the tags "real and AI-generated." The format in which each of the video files will be tagged will be by creating a CSV file where each video file will correspond to label depending on the folder it was stored. These include filename and label, which in this case are either 'real' or 'AI', indications that the dataset to be produced, maintains its usability for training, devoid of the inconveniences of wrong labels by a human being.

2. Preprocessing

   The final preparation before feeding into the model is referred to as preprocessing of the video data. The computer vision tool called OpenCV library will be used to split each of the mentioned videos into frames . To achieve a temporal consistency in all the recordings, Every 1 frame gets capture from 3 frames and continue until all the frames are not extracted from the video. This is important as fluctuation in the rate of frames in a video may introduce some unpredictable nature that interferes with the operation of the model.

   After frame extraction, the pictures will be resized to 224 by 224 pixels, this is because of MobileNetV2 and InceptionV3 prefer an image size of 224 by 224 pixels. As a result, this scaling does not increase the computational load and ensures that the frames can be placed within the architectural framework of the models. Basically, in a deep learning model, the pixel is normalized and adopts a standard range of [0,1]. In doing so, issues likely to occur due to variance in either the brightness or contrast of frames are also done away with.

   It is noteworthy that optical flow calculation is an essential part of the preprocessing stage. Inter-frame motion is obtained using optical flow which provides valuable temporal data about the movement and behaviors within the videos. This is especially useful in videos where these aberrations may be present due to the poor creation methodology, where AI is used to produce videos.

3. Model Architecture

   The merging of MobileNetV2 and InceptionV3 for spatial feature extraction forms the basis of the suggested system. Because of their complementing strengths, these

models were selected: InceptionV3 is exceptionally good at capturing complex information in the data, whereas MobileNetV2 provides computational efficiency.

[A] MobileNetV2 for Efficient Feature Extraction -

By virtue of its compactness and run-time capability, MobileNetV2 is chosen as one of feature extractors in this architecture, which is ideal for application, for instance, video tracking and monitoring. MobileNetV2 employs depthwise separable convolutions, which divide the convolution process into two components: point-wise convolution which combines depths of depth-wise convolution and depth-wise convolution which passes one convolutional filter through the input channels. In contrast with conventional convolution layers, this reduces the number of parameters and computations in order that the model can process video frames fast.

This innovation by MobileNetV2 is another called the inverted residual block with a linear bottleneck. Hence it is named as an inverted residual block since it first compresses the shape of the input features before going for expansion, while most residual blocks force the shape of the block's dimensionality before compressing features. This design also provides a reduction in parameters while maintaining spatial information. For MobileNetV2 to be able to capture rich textures, edge, and other site-dependent features in every frame without using too much resources, there is a bottleneck layer that ensures these features are preserved.

Interestingly, MobileNetV2 is beneficial when analyzing hundreds of frames from a video. It is very useful for quickly and accurately determining the content of each frame owing to its ability to selectively find essential spatial attributes, thereby minimising computational load.

[B] InceptionV3 for Advanced Detail Recognition -

To enhance the capacity of the proposed model for detecting subtleties of high-level features InceptionV3 is incorporated into the architecture. InceptionV3 contains inception modules that perform multiple convolution operations simultaneously, one of which includes filters of size 1x1, 3x3, or 5x5 through which a network tries to make sense of the image at multiple scales. Because of this architecture, InceptionV3 can keep track of more delicate structures in the video frames apart from what other local peculiarities such as texture and edges. InceptionV3 in a position to recognising fine features as well as large contextual information due to the many filters that it applies in parallel.

It is important to mention here that this structure employs factorised convolutions, which break a significantly large convolution into many little, more efficient ones.

For example, a 3x3 convolution can be separated into two successive operations – 1x3 and 3x1, which can reduce the amount of parameters to be learned without storing the representational power needed to learn spatial hierarchies. Due to this factorisation, the network can maintain a high level of functioning on the network while not affecting the computing overhead.

Furthermore, InceptionV3 architecture incorporates auxiliary classifiers in training that introduces side supervision and makes the improvement upon the gradient vanishment problem in deep learning architectures. These classifiers help in adjusting steep gradients with deeper layers to improve learning to ensure that a model

achieves convergence rapidly and accurately would be largely benefited by these classifiers.

An exhaustive description of each frame is derived from the properties obtained from each frame of the video by applying MobileNetV2 and InceptionV3. The Recurrent Neural Network (RNN), the following step in the architecture, takes those properties and use them to remember temporal correlations with frame. The RNN plays the most important role in detecting temporal anomilies that are so characteristic for movies created by AI, like motion change acceleration or weird switching.

4. Training

   To train the model, binary cross entropy is good to go for this binary classification problem (actual vs generated by AI). In the purpose of training, the Adam optimiser together with the learning rate scheduler will be used so that the rate of learning will change based on the training progress. This ensures that from the data fed into the model the best is extracted and attempt to prevent over fitting is made.

   In data division, a proportion of 20 percentage of the given dataset will be used for testing and the other 80 percentage for training. This type of training ensures that model is tested on data it has never seen before thus giving as a true measure of the ability to generalize.

5. Evaluation Metrics

   Even though accuracy, and precision, recall, F1- score are standard measures of the model's performance, they will be utilized in this paper. The metrics include false positivity, which is when a real video is classified as an AI generated one, or false negativity which is when an AI generated video is not detected at all will help in determining the models ability to sort videos between the two categories.

   The graphical representation of true positives, true negatives, false positives and false negatives using the confusion matrix will also be presented. When both InceptionV3 and MobileNetV2 are used, the model should have ideal performance across these since InceptionV3 possesses the advantage of deep and multi-scale analysis and MobileNetV2 has an effective feature extraction.

6. Front-End Interface

   Integrating the learnt model into an intuitive front-end interface is the project's last phase. Users will be able to post videos using this interface and get immediate feedback on whether the video is artificial intelligence (AI)-generated or real. Frameworks like Flask will be used to put the model on a server, allowing it to effectively handle user requests.

   HTML, CSS, and Flask Framework will be used to create the front-end, which will give users an easy-to-use interface. After a video is uploaded, the model will process it and show the outcome on the screen. InceptionV3 and MobileNetV2 work together to guarantee that the model can produce precise and timely results, even in real-time situations.

   The model will be tuned to process frames effectively, utilising the speed of MobileNetV2 and the high-level detail identification of InceptionV3 to manage video uploads seamlessly. This makes feedback almost instantaneous, giving users a smooth experience.

Chapter 4

System Implementation

4.1 MobileNetV2

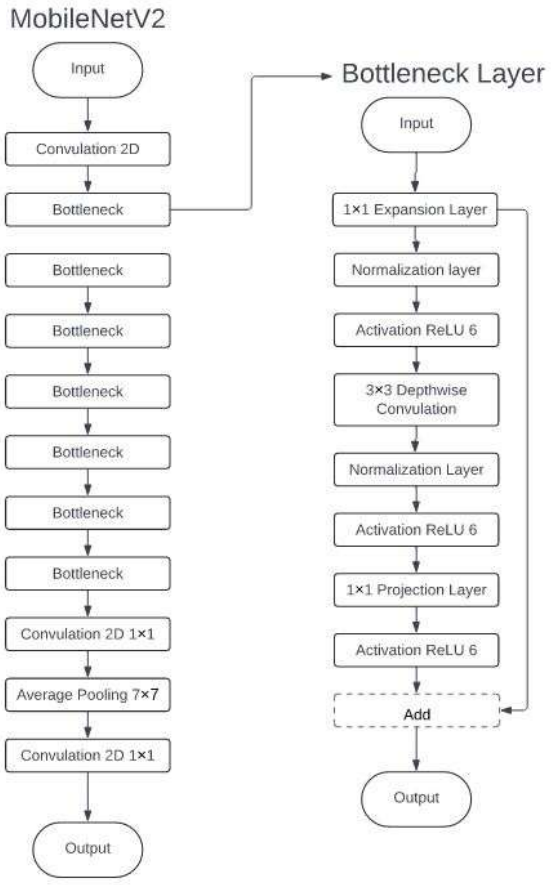

Figure 4.1: MobileNetV2

4.1.1 MobileNetV2

1. Input Layer - This layer receives the input image, typically resized to a shape like (224x224x3) for 224x224 pixels and 3 color channels (RGB). The following layer processes this input.

2. Convolution 2D Layer - This is a standard 2D convolutional layer applied to the input. It extracts low-level characteristics like edges, corners, and textures using a 3x3 kernel size, which is common in convolutional networks. - The depthwise separable technique is used by all other layers of the network, with this being the only conventional convolutional layer. The objective is to extract the image's basic spatial information.

3. Multiple Bottleneck Layers - The fundamental components of MobileNetV2 are these bottleneck layers, which are stacked one on top of the other. - As seen in the first illustration, the bottleneck layer is made up of the following: 1x1 Expansion Layer: increases the dimensions by expanding the input channels by a factor (e.g., 6x), which enables the extraction of additional features. This is significant since MobileNetV2 saves computational expense by using fewer channels, therefore temporarily enlarging them aids in the extraction of significant patterns. The depthwise convolution (3x3) is as follows: Compared to normal convolution, a depthwise convolution significantly lowers computational costs by applying a 3x3 filter to each channel independently. It treats each channel independently as opposed to applying a kernel to all of the channels at once. One-by-one projection layer: As a result, the anticipated number of channels is reduced to the initial size of the input channel. It lessens redundancy and aids in condensing the information after processing. Residual/Skipping Connection: A residual connection is added if the dimensions of the input and output are identical. By learning the difference between input and output (rather than the output itself), the model can converge more quickly and prevent vanishing gradients thanks to this connection.

   The Bottleneck Layer's Objective: - Combines depthwise separable convolutions with pointwise convolutions to lower the overall computational strain. By use of the residual connection, crucial information is preserved. - For effective feature learning, combines compressed representation with enlarged feature space.

4. Convolution 2D 1x1 - Following the bottleneck layers, an additional 1x1 convolutional layer is implemented to aid in dimension compression following several layers of bottleneck operations. Aim: After processing several features in the preceding layers, reduce the dimensionality.

5. Average Pooling Layer (7x7) - This layer, which is a global average pooling layer, averages over all spatial dimensions (7x7 regions) to reduce each feature map to a single value. In order to flatten the output for the final classification layer, each feature map's spatial dimensions should be reduced to 1x1.

6. Convolution 2D 1x1 (Final Layer) - For instance, if ImageNet contains 1000 classes, this layer will decrease the feature map to 1000 channels. This last 1x1 convolutional layer is used to match the number of classes in the output. The objective is to generate a final output that corresponds to the total number of classes that can be found in the classification work.

7. Output Layer - The network generates its predictions at this last layer. In the event that the task involves classification, the output logits are transformed into probabilities using a softmax activation function. The objective is to provide the final classification prediction, indicating whether or not the image belongs to a specific class.

4.1.2 Bottleneck Layer

1. Expansion Layer 1x1 The input's channel count is increased by this layer. For instance, a 1x1 convolution can be used to increase an input with 32 channels to 192 channels. Goal: In order to enable greater feature extraction by later convolutions, this expansion raises the dimensionality. More abstract features can be captured by setting the expansion factor to a constant value, such as 6.

2. Normalization Layer- The data goes through a normalization layer (usually batch normalization) following the expansion. Goal: By normalizing the outputs of the preceding layer, batch normalization lowers the possibility of vanishing or exploding gradients and ensures that activations have a regular range of values, stabilizing and speeding up training.

3. ReLU6 Layer Activation - ReLU6 a variant of the conventional ReLU activation (Rectified Linear Unit), is the activation function utilized here; however, it clips the values up to a maximum of 6. - Goal The model is more effective for low-precision calculations when ReLU6 is used because it enhances quantization performance on mobile devices.

4. 3x3 Depthwise Convolution - The bottleneck layer uses depthwise convolutions in place of conventional 3x3 convolutions. Instead of convolving across all channels, depthwise convolution convolves each input channel independently. Goal: MobileNetV2 is very efficient since depthwise convolutions drastically cut down on computation and the amount of parameters. It enables the extraction of spatial features at a reduced computational cost.

5. Normalization Layer - Following the depthwise convolution, a second normalization layer is applied, this one normalizing the data once more to guarantee that activations behave properly for the subsequent layers. Goal: To preserve steady gradients and enhance convergence, this is required.

6. Activation ReLU6 Layer - Following normalization, another ReLU6 activation is applied. This facilitates non-linear transformations and aids the network in discovering intricate patterns within the input.

7. 1x1 Projection Layer - A 1x1 projection convolution is used to return the number of channels to the original input dimension following processing of the enlarged data. For instance, this layer projects it back to 32 channels after it has expanded from 32 to 192 channels. Goal: The next layer can process the compressed representation of the expanded data thanks to the assistance of this projection phase.

8. Residual Connection (Add) - A residual connection is added, in which the input is added directly to the output, provided that the input and output dimensions

are equal. Goal: Instead of learning the output directly, this connection teaches the network to distinguish between input and output. By permitting gradient flow across layers, this residual technique facilitates faster learning and stabilizes deep network training.

9. Output - The bottleneck layer generates its final output, which is sent to the following layer, following the residual connection, if any. Goal: to generate a collection of condensed, non-redundant features that have been improved and extracted using different activations, normalizations, and convolutions.

4.2 InceptionV3

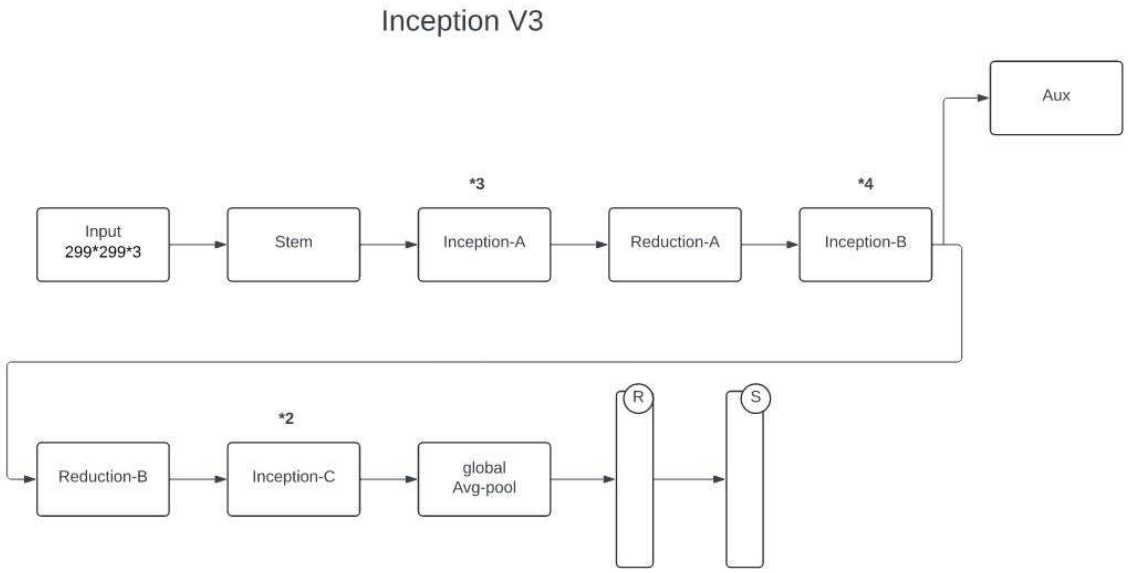

Figure 4.2: InceptionV3

1. Input Layer (299*299*3) To represent an RGB image, Inception V3's input size is set to 299x299 with three channels. This image is processed using many pooling and convolutional layers.

2. Stem The first block to extract fundamental information from the input is called the stem. Convolutional layers are typically included, followed by batch normalisation and ReLU activation. The stem increases the image's depth (number of feature maps) while decreasing its spatial dimensions.

3. Inception-A Block (×3) A module called the Inception-A block is made to record features at various scales. It has convolutions with varying kernel sizes across multiple branches. Different kinds of information are captured by each branch, and the output is concatenated along the depth dimension. To improve feature extraction at this stage, this block is performed three times.

4. Reduction-A Block The Reduction-A block preserves or increases the depth of the feature maps while decreasing their spatial dimensions. Usually, a mix of pooling procedures and convolutions is used for this. Downsampling, lowering computational cost, and allowing the network to concentrate on more abstract information are all made possible by this block.

5. Inception-B Block ($\times 4$) The network next enters the Inception-B block, which again has several branches that capture various feature scales, following the Reduction-A block. The features recorded in the prior stages are further refined by repeating the Inception-B block four times.

6. Auxiliary Classifier To avoid vanishing gradients during training, an auxiliary classifier is included. A fully connected layer and a few convolutional layers make up this system, which is utilised for intermediate supervision during training and produces an early prediction. This facilitates the network's regularisation and enhances its convergence.

7. Reduction-B Block The Reduction-B block, like Reduction-A, lowers the spatial dimensions once again in order to get ready for the network's deeper levels. The network may concentrate on more abstract and high-level aspects thanks to this block, which makes sure that the number of parameters stays manageable.

8. Inception-C Block ($\times 2$) Despite having differing convolutional configurations and kernel sizes, the Inception-C block functions similarly to the Inception-A and Inception-B blocks. It helps to collect a wider variety of feature types because it is replicated twice in the network.

9. Global Average Pooling Global average pooling is used to average the feature maps across the spatial dimensions following the last Inception block. As a result, each feature map is reduced to a single value that indicates whether the feature is present throughout the entire image. By preventing overfitting, global average pooling drastically lowers the number of parameters.

10. Fully Connected Layers (R and S) The output is transmitted through one or more fully linked (dense) layers (designated as R and S in the picture) following the global average pooling. The number of units in the final layer corresponds to the number of classes in the dataset, and these layers generate the final classification. In order to generate class probabilities, a softmax activation function is typically employed at the conclusion.

Chapter 5

Result and Analysis

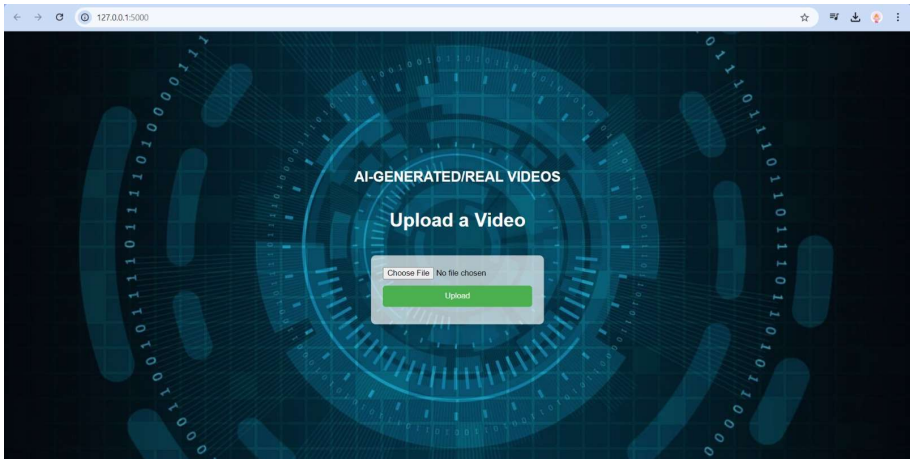

Figure 5.1: Homepage - This is the homepage of the website.

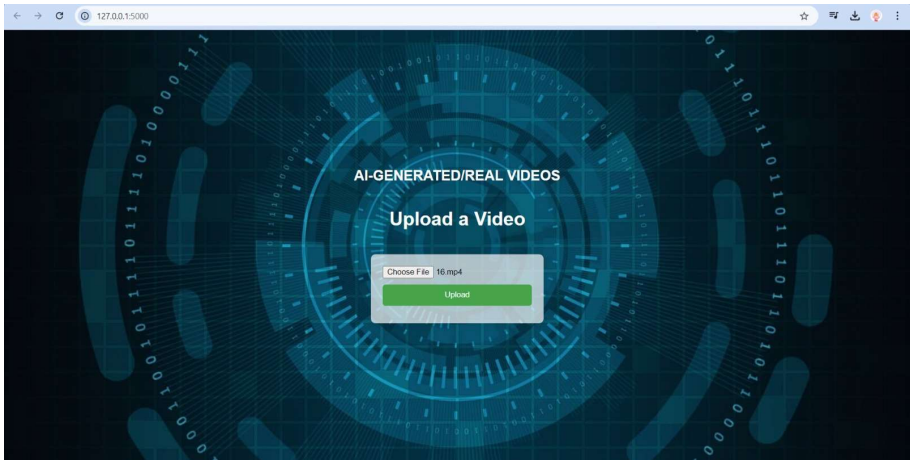

Figure 5.2: Input - This is where an AI generated video is selected to upload on the website for the model to work and detect whether it is a real or AI generated video

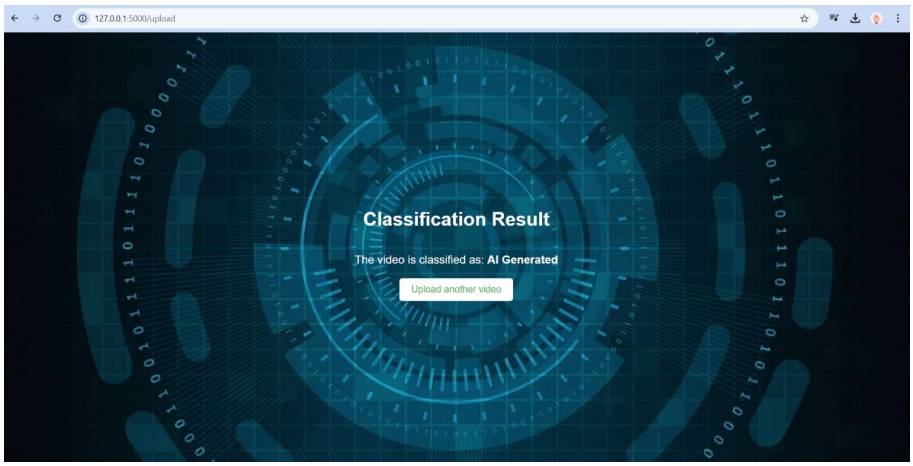

Figure 5.3: Output - The model classifies the uploaded video as AI generated.

```
1342/1342 ━━━━━━━━━━━━━━━━━━━━ 3s 2ms/step - accuracy: 0.7752 - loss: 0.6214 - val_accuracy: 0.9666 - val_loss: 0.0861
Epoch 2/20
1342/1342 ━━━━━━━━━━━━━━━━━━━━ 2s 1ms/step - accuracy: 0.8845 - loss: 0.3324 - val_accuracy: 0.9091 - val_loss: 0.1672
Epoch 3/20
1342/1342 ━━━━━━━━━━━━━━━━━━━━ 2s 1ms/step - accuracy: 0.8457 - loss: 0.4235 - val_accuracy: 0.9151 - val_loss: 0.1845
Epoch 4/20
1342/1342 ━━━━━━━━━━━━━━━━━━━━ 2s 1ms/step - accuracy: 0.8638 - loss: 0.4063 - val_accuracy: 0.8944 - val_loss: 0.2135
Epoch 5/20
1342/1342 ━━━━━━━━━━━━━━━━━━━━ 2s 1ms/step - accuracy: 0.8544 - loss: 0.3798 - val_accuracy: 0.8915 - val_loss: 0.2487
Epoch 6/20
1342/1342 ━━━━━━━━━━━━━━━━━━━━ 2s 2ms/step - accuracy: 0.8395 - loss: 0.4832 - val_accuracy: 0.8877 - val_loss: 0.2233
Epoch 7/20
1342/1342 ━━━━━━━━━━━━━━━━━━━━ 2s 1ms/step - accuracy: 0.8483 - loss: 0.4513 - val_accuracy: 0.8966 - val_loss: 0.1831
Epoch 8/20
1342/1342 ━━━━━━━━━━━━━━━━━━━━ 2s 2ms/step - accuracy: 0.8417 - loss: 0.4399 - val_accuracy: 0.8996 - val_loss: 0.1835
Epoch 9/20
1342/1342 ━━━━━━━━━━━━━━━━━━━━ 2s 1ms/step - accuracy: 0.8484 - loss: 0.4412 - val_accuracy: 0.9040 - val_loss: 0.1857
Epoch 10/20
1342/1342 ━━━━━━━━━━━━━━━━━━━━ 2s 1ms/step - accuracy: 0.8411 - loss: 0.4448 - val_accuracy: 0.8862 - val_loss: 0.2176
Epoch 11/20
1342/1342 ━━━━━━━━━━━━━━━━━━━━ 2s 1ms/step - accuracy: 0.8505 - loss: 0.4598 - val_accuracy: 0.9050 - val_loss: 0.1965
Epoch 12/20
1342/1342 ━━━━━━━━━━━━━━━━━━━━ 3s 2ms/step - accuracy: 0.8533 - loss: 0.4659 - val_accuracy: 0.8944 - val_loss: 0.2444
Epoch 13/20
1342/1342 ━━━━━━━━━━━━━━━━━━━━ 2s 1ms/step - accuracy: 0.8548 - loss: 0.4521 - val_accuracy: 0.8826 - val_loss: 0.2726
Epoch 14/20
1342/1342 ━━━━━━━━━━━━━━━━━━━━ 2s 1ms/step - accuracy: 0.8309 - loss: 0.4606 - val_accuracy: 0.8903 - val_loss: 0.2149
Epoch 15/20
1342/1342 ━━━━━━━━━━━━━━━━━━━━ 2s 1ms/step - accuracy: 0.8600 - loss: 0.4392 - val_accuracy: 0.9099 - val_loss: 0.2353
Epoch 16/20
1342/1342 ━━━━━━━━━━━━━━━━━━━━ 2s 2ms/step - accuracy: 0.8791 - loss: 0.4023 - val_accuracy: 0.9121 - val_loss: 0.1954
Epoch 17/20
1342/1342 ━━━━━━━━━━━━━━━━━━━━ 2s 1ms/step - accuracy: 0.8747 - loss: 0.4441 - val_accuracy: 0.8984 - val_loss: 0.2059
Epoch 18/20
1342/1342 ━━━━━━━━━━━━━━━━━━━━ 2s 2ms/step - accuracy: 0.8648 - loss: 0.4087 - val_accuracy: 0.8812 - val_loss: 0.3160
Epoch 19/20
1342/1342 ━━━━━━━━━━━━━━━━━━━━ 2s 2ms/step - accuracy: 0.8658 - loss: 0.4697 - val_accuracy: 0.9064 - val_loss: 0.2284
Epoch 20/20
1342/1342 ━━━━━━━━━━━━━━━━━━━━ 2s 1ms/step - accuracy: 0.8682 - loss: 0.4011 - val_accuracy: 0.9042 - val_loss: 0.2192
420/420 ━━━━━━━━━━━━━━━━━━━━ 0s 629us/step - accuracy: 0.9026 - loss: 0.2201
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is considered legacy. We
recommend using instead the native Keras format, e.g. `model.save('my_model.keras')` or `keras.saving.save_model(model, 'my_model.keras')`.
Test Accuracy: 0.8992
```

Figure 5.4: Testing Phase - Testing of the dataset.

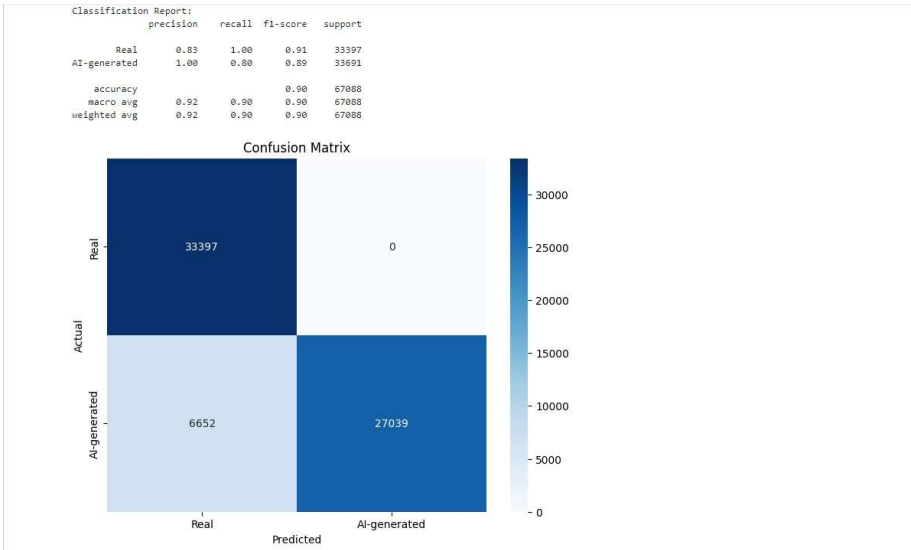

Figure 5.5: Accuracy Metrics - Various Metrics displaying their values.

Figures 5.1 5.2 5.3, Display the front-end of the model that we created using our own 'AI-Generated — Real videos' dataset. It is a simple Web application created using Flask that accepts a video as an input and classifies it as AI-Generated or Real.

As shown in Figure 5.4, The model performed well in learning to distinguish between real and AI-generated videos. From the beginning, it demonstrated excellent generalisation skills by performing admirably on new, unseen data. The model improved its accuracy on training data over the course of 20 training epochs, ultimately reaching 90% accuracy overall. Its 90% confidence level in predictions for the validation set held steady throughout time, despite some decline. Although the model's high final loss number indicates that there may be some particularly challenging examples in the dataset, overall it was successful in learning to distinguish between real and AI-generated videos.

As shown in Figure 5.5, The confusion matrix proves that The Model has successfully predicted all the actual real videos as real and predicted about 80.25% actual AI-generated videos as AI- Generated. This goes on to show that it is comparatively difficult for our model to recognize Actual AI-Generated videos than Real videos due to very close similarities.

Chapter 6

Conclusion and Future scope

This work successfully demonstrates the utilization of both spatial and temporal feature extraction a CNN-RNN system that enables the discrimination of real videos from AI-synthesized ones. The system is able to accurately distinguish between real and fake intelligence generated videos by integrating the use of CNNs in frame-based features and RNNs in temporal features.The strategy recommended positive results through performance indicators such as recall, accuracy together with the confusion matrix, f-1 score and precision.een real and AI-generated videos. The system successfully identifies important distinctions between real and artificial intelligence-generated videos by fusing the capabilities of CNNs for frame-based analysis and RNNs for temporal sequence learning. The strategy produced encouraging outcomes, as seen by high performance indicators including recall, accuracy, confusion matrix, f-1 score and precision. That the model is extremely useful for video integrity monitoring in real-time while watching video again in authentic online interface confirms the potency of the proposed solution in preventing AI-fake news circulation.

Future research can assess the generalization of the present findings to a larger sample of replenished AI-generated content, such as deepfakes, videos created by sophisticated generative models, and any other productions of synthetic media. The performance can also be enhanced by incorporating state-of-art structures such as transformers or other attention based methods to improve the generalization capacity of the model. For enhanced accessibility, optimisers for real-time analysis and the implementation of mobile interfaces might also be developed. Another way of gaining insight into the decision-making process could be investigating which of the AI methods can be applied to a specific system, making the system more transparent and thus more believable to the end user.

References

[1] Bai, Jianfa Lin, Man Cao, Gang. (2024). AI-Generated Video Detection via Spatio-Temporal Anomaly Learning.

[2] Gazi Hasin Ishrak, Zalish Mahmud, MD. Zami Al Zunaed Farabe, Tahera Khanom Tinni, Tanzim Reza, Mohammad Zavid Parvez. (2024) Explainable Deepfake Video Detection using Convolutional Neural Network and CapsuleNet.

[3] Hossain, Md. Zahid Uzzaman, Farhad Islam, Md. Rakibul. (2023). Advancing AI-Generated Image Detection: Enhanced Accuracy through CNN and Vision Transformer Models with Explainable AI Insights.

[4] Al-Adwan, Aryaf Alazzam, Hadeel Al-Anbaki, Noor Alduweib, Eman. (2024). Detection of Deepfake Media Using a Hybrid CNN–RNN Model and Particle Swarm Optimization (PSO) Algorithm.

[5] S. R. Adnan and H. A. Abdulbaqi, "Deepfake Video Detection Based on Convolutional Neural Networks," 2022 International Conference on Data Science and Intelligent Computing (ICDSIC), Karbala, Iraq, 2022.

[6] Y. Al-Dhabi and S. Zhang, "Deepfake Video Detection by Combining Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN)," 2021 IEEE International Conference on Computer Science, Artificial Intelligence and Electronic Engineering (CSAIEE), SC, USA, 2021.

[7] I. Amerini, L. Galteri, R. Caldelli and A. Del Bimbo, "Deepfake Video Detection through Optical Flow Based CNN," 2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW), Seoul, Korea (South), 2019.

[8] D. Güera and E. J. Delp, "Deepfake Video Detection Using Recurrent Neural Networks," 2018 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), Auckland, New Zealand, 2018,

[9] S. Agarwal, S., et al. Detecting deep-fake videos from appearance and behavior. IEEE international workshop on information forensics and security (WIFS), IEEE. 2020

[10] Khormali, A. and J.-S. Yuan "ADD: Attention-Based DeepFake Detection Approach." Big Data and Cognitive Computing 5(4): 49. (2021).

[11] A. Shende, "Using deep learning to detect deepfake videos." Turkish Journal of Computer and Mathematics Education (TURCOMAT) 12(11): 5012-5017. (2021).

[12] A. V. Nadimpalli, and A. Rattani On improving cross-dataset generalization of deepfake detectors. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. (2022).

[13] H. F.Shahzad, et al. "A Review of Image Processing Techniques for Deepfakes." Sensors 22(12): 4556. (2022).

[14] T. T. Nguyen,., et al"Deep learning for deepfakes creation and detection: A survey." arXiv preprint arXiv:1909.11573. . (2019).

[15] D. Yadav, and S. Salmani Deepfake: A survey on facial forgerytechnique using generative adversarial network. 2019 Internationalconference on intelligent computing and control systems (ICCS), IEEE. (2019).

[16] Rahman, A., et al. (2022). "A Qualitative Survey on Deep Learning Based Deep fake Video Creation and Detection Method." Aust. J. Eng. Innov. Technol 4(1): 13-26.

[17] A. Ismail, et al. "Deepfake video detection: YOLO-Face convolution recurrent approach." PeerJ Computer Science 7: e730. (2021).

[18] B. Dolhansky, J. B. ton, B. Pflaum, J. Lu, R. Howes, M. Wang and C. C. Ferrer, O., 2020. "The DeepFake Detect ion Challenge (DFDC) Dataset". Available at https://ai.facebook.com/datasets/dfdc.

[19]] G. Lee, and M. Kim "Deepfake Detection Using the Rate of Change between Frames Based on Computer Vision." Sensors 21(21): 7367. (2021).

[20]] L. Guarnera, et al. Deepfake detection by analyzing convolutional traces. Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops. (2020).

[21] de Lima, O., et al. "Deepfake detection using spatiotemporal convolutional networks." arXiv preprint arXiv:2006.14749. (2020).

[22] A. Khodabakhsh, et al. A taxonomy of audiovisual fake multimedia content creation technology. 2018 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR), IEEE. (2018).

[23] A. Sathiya Priya and T. Manisha. (2024). CNN and RNN using Deepfake detection.

[24] Li, Y. and S. Lyu "Exposing deepfake videos by detecting face warping artifacts." arXiv preprint arXiv:1811.00656. (2018).

[25] https://soravideos.media/

[26] https://pixabay.com/videos/search/real/