DEEPFAKE GENERATION AND DETECTION USING DEEP LEARNING

PROJECT REPORT

submitted by

AGHEEL KAREEM (SCM18CS005)

AMAL HAFIZ (SCM18CS011)

ARYAN C R (SCM18CS020)

GRACIOUS BABY (SCM18CS035)

to

The APJ Abdul Kalam Technological University

in partial fulfilment of the requirements for the award of the Degree

of

Bachelor of Technology

In

Computer Science Engineering

Department of Computer Science Engineering

SCMS School of Engineering and Technology

Vidya Nagar, Karukutty, Kerala-683576

JUNE 2022

DECLARATION

I undersigned hereby declare that the project report (“DEEP FAKE GENERATION AND DETECTION USING DEEP LEARNING”), submitted for partial fulfilment of the requirements for the award of the degree of Bachelor of Technology of the APJ Abdul Kalam Technological University, Kerala is a bonafide work done by me under supervision of
Assoc. Prof Sonal Ayyappan. This submission represents our ideas in our own words and where ideas or words of others have been included, we have adequately and accurately cited and referenced the original sources. We also declare that we have adhered to academic honesty and integrity ethics and have not misrepresented or fabricated any data, idea, fact, or source in our submission. We understand that any violation of the above will be a cause for disciplinary action by the institute and/or the University and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been obtained. This report has not previously formed the basis for awarding any degree, diploma, or similar title to any other University.

Place: Karukutty, Ernakulam Dist

Date: 27th June 2022

Signature & Name of the Students

AGHEEL KAREEM

AMAL HAFIZ

ARYAN C R

GRACIOUS BABY

DEPARTMENT OF COMPUTER SCIENCE ENGINEERING

SCMS SCHOOL OF ENGINEERING AND TECHNOLOGY, KARUKUTTY

CERTIFICATE

This is to certify that report entitled "**DEEP FAKE GENERATION AND DETECTION USING DEEP LEARNING"** is a bonafide record of the project presented by Agheel Kareem, Amal Hafiz**, Aryan CR**, Gracious Baby to the APJ Abdul Kalam Technological University in partial fulfilment of the requirement or the award of B.Tech degree in Electronics and Communication Engineering is a bonafide record of the project carried out by him/her under my /our guidance and supervision. This report in any form has not been submitted to any other University or Institute for any purpose.

Internal Supervisor	External Examiner	HEAD OF THE DEPARTMENT Dr. Varun G Menon

ACKNOWLEDGEMENT

We thank God Almighty for the blessings which helped us throughout this humble endeavour. We sincerely acknowledge our gratitude to the Principal of SCMS School of Engineering and Technology, **Prof. Praveensal C J**, for granting us all the resources available in the college to make our project. We also extend our gratitude to the head of the Department of Computer Science Engineering **Associate Prof. Varun G Menon,** for the valuable support throughout the project. This project would not have been a success if not for the timely advice, guidance, and support of our project guide and project coordinator**, Associate Prof. Sonal Ayyappan,** Computer Science Engineering Department, and all the Computer Science Engineering Department faculties. We would also like to express our gratitude to the lab assistants for their valuable support throughout the work on this project. We thank all our friends whose help and support were indispensable for the successful completion of our project.

ABSTRACT

Modern technologies like Tensorflow or Keras, open-source trained models, and affordable computer infrastructure have made it easy to swiftly generate deepfakes to propagate disinformation, revenge porn, financial fraud, hoaxes, and disrupt government processes.

The primary areas of current study have been deep fake picture and video detection. Deepface algorithms have the potential to create fake images and videos that are difficult to identify from real ones. The development of tools that can quickly recognise and assess the integrity of digital visual material is essential.

Deep learning algorithms have grown so potent due to increased computing power that it is now relatively easy to produce human-like synthetic videos, known as "deep fakes." One may readily imagine scenarios in which these realistic face-switched deep fakes are used to incite political unrest, stage terrorist attacks, produce revenge pornography, and extort individuals. In this paper, we provide a novel deep learning-based strategy for the efficient separation of fraudulent films produced by AI from actual ones. Our technique is able to recognize the replacement and recreation of deep fakes automatically. We are trying to use Artificial Intelligence to compete against Artificial Intelligence (AI).

The frame-level characteristics are extracted by our system using a Res-Next Convolution neural network, and these features are then used to train an LSTM-based recurrent neural network (RNN) to determine if the video has been altered in any way or not, i.e. whether it is a deep fake or authentic video. We test our technique using a sizable quantity of balanced and mixed data sets created by combining the different accessible data sets, such as Celeb-DF[3], in order to simulate real-world circumstances and improve the model's performance on real-world data. We also demonstrate a very straightforward and reliable method for how our system might produce competitive results.

The majority of current co-part segmentation techniques work in supervised learning environments, which demand a lot of annotated data for training. We provide a self-supervised deep learning strategy for co-part segmentation to get over this restriction. In contrast to other efforts, our method advances the notion that usable object components may be found by using motion information inferred from movies. As a result, our technique makes use of pairs of frames that were taken from the same video. In order to rebuild the target picture, the network learns to forecast component segments together with a representation of the motion between two frames. We show that our technique can yield better segmentation maps than earlier self-supervised co-part segmentation algorithms through comprehensive experimental assessment on publically available video sequences.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1
INTRODUCTION

In the world of ever growing Social media platforms, Deep Fakes are considered as the major threat of AI. There are many Scenarios where these realistic face swapped deepfakes are used to create political distress, fake terrorism events, revenge porn, blackmail peoples are easily envisioned.Some of the examples are Brad Pitt, Angelina Jolie nude videos. It becomes very important to spot the difference between the deepfake and pristine video.

1.1 DETECTION OF DEEP FAKE

We are using AI to fight AI.Deep Fakes are created using tools like FaceApp[2] and Face Swap [3], which use pre-trained neural networks like GAN or Auto encoders for these deep fakes creation. Our method uses a LSTM based artificial neural network to process the sequential temporal analysis of the video frames and pre-trained Res-Next CNN to extract the frame level features. ResNext Convolution neural network extracts the frame-level features and these features are further used to train the Long Short Term Memory based artificial Recurrent Neural Network to classify the video as Deep Fake or real. To emulate the real time scenarios and make the model perform better on real time data, we trained our method with a large amount of balance and combination of various available dataset like Celeb-DF[1]. Further to make it ready to use for the customers, we have developed a front end application where the user will upload the video. The video will be processed by the model and the output will be rendered back to the user with the classification of the video as deep fake or real and confidence of the model.

The increasing sophistication of mobile camera technology and the ever growing reach of social media and media sharing portals have made the creation and propagation of digital videos more convenient than ever before. Deep learning has given rise to technologies that would have been thought impossible only a handful of years ago. Modern generative models are one example of these, capable of synthesising hyper realistic images, speech, music, and even video. These models have found use in a wide variety of applications, including making the world more accessible through text-to-speech, and helping generate training data for medical imaging.

Spreading of the Deep fakes over the social media platforms have become very common leading to spamming and speculating wrong information over the platform. Just imagine a deep fake of our prime minister declaring war against neighbouring countries, or a Deep fake of a reputed celebrity abusing the fans. These types of deep fakes will be terrible, and lead to threatening, misleading common people. To overcome such a situation, Deep fake detection is very important. So, we describe a new deep learning-based method that can effectively distinguish AIgenerated fake videos (Deep Fake Videos) from real videos. It's incredibly important to develop technology that can spot fakes, so that the deep fakes can be identified and prevented from spreading over the internet.

However there are a few constraints for this:

•User: Users of the application will be able detect whether the uploaded video is fake or real, Along with the model confidence of the prediction.

• Prediction: The User will be able to see the playing video with the output on the face along with the confidence of the model.

• Easy and User-friendly User-Interface: Users seem to prefer a more simplified process of Deep Fake video detection. Hence, a straightforward and user-friendly interface is implemented.The UI contains a browse tab to select the video for processing. It reduces the complications and at the same time enrich the user experience.

• Cross-platform compatibility: with an ever-increasing target market, accessibility should be your main priority. By enabling a cross-platform compatibility feature, you can increase your reach to across different platforms. Being a server side application it will run on any device that has a web browser installed in it.

Now during the detection of Deepfake , our model turns the video into frames which is then analysed.

The model identifies certain segments or parameters from the frames. These include Blinking of eyes, Teeth enchantment , Bigger distance for eyes , Moustaches , Double edges, eyes, ears, nose , Iris segmentation , Wrinkles on face , Inconsistent head pose , Face angle , Skin tone , Facial Expressions , Lighting , Different Pose , Double chins , Hairstyle , Higher cheekbones.

After analysis we decided to use the PyTorch framework along with python3 language for programming. PyTorch is chosen as it has good support for CUDA i.e Graphic Processing Unit (GPU) and it is customise-able. Google Cloud Platform for training the final model on large number of data-sets

We evaluated our model with a large number of real time dataset which include YouTube videos dataset. Confusion Matrix approach is used to evaluate the accuracy of the trained model.

1.2 GENERATION OF DEEP FAKE

We also Create a model for the generation of deep fake videos through the concept of Motion Cosegmentation. Discovering objects and object parts in images is one of the fundamental steps towards semantic understanding of visual scenes. In computer vision this problem is referred to as semantic segmentation and is approached within a machine learning framework as a dense labelling task, where the goal is to assign a categorical label to each pixel of an image. In this paper we address a more challenging problem and a special case of semantic segmentation, referred to as co-part segmentation. The task of the co-part segmentation problem is to identify segments corresponding to different parts within a single object. For instance, in a human body the relevant parts correspond to hands, legs, head and torso. Such parts are of special interest for the automatic analysis of visual scenes since they constitute intermediate representations, which are robust to sensor changes and appearance variations.Discovering objects and object parts in images is one of the fundamental steps towards semantic understanding of visual scenes. In computer vision this problem is referred to as semantic segmentation and is approached within a machine learning framework as a dense labelling task, where the goal is to assign a categorical label to each pixel of an image. In this paper we address a more challenging problem and a special case of semantic segmentation, referred to as co-part segmentation. The task of the co-part segmentation problem is to identify segments corresponding to different parts within a single object. For instance, in a human body the relevant parts correspond to hands, legs, head and torso. Such parts are of special interest for the automatic analysis of visual scenes since they constitute intermediate representations, which are robust to sensor changes and appearance variations.

Recently, co-part segmentation algorithms have gained popularity as they are key-enabling components for image editing and animations tools. Several works use automatically computed object parts for virtual try-on [4], pose-guided generation [5], face-swap and re-enactment [6]. The vast majority of co-part segmentation methods operates in a supervised setting [7], thus requiring a large amount of human annotated data. To overcome this limitation some works have focused on the challenging problem of unsupervised co-part segmentation [8], demonstrating that object parts can be inferred without relying on labelled data. While effective, these approaches are inherently limited by the fact that only appearance-based features are exploited to compute segments. In contrast we argue and experimentally demonstrate that more precise information about object parts can be extracted by leveraging motion information from vast amounts of unlabeled videos. Thus, in this paper we propose a novel video-based self-supervised co-part segmentation method. Once trained, our proposed framework takes a single image as input and outputs the segmentation map indicating different object parts. Differently from previous methods [8], we assume that at training time, a large collection of videos depicting objects within the same category is available.

Our strategy draws inspiration from earlier studies on self-supervised key point estimation [9]. These techniques' principal goal is to separate an object's semantic and visual representations by imposing a reconstruction target. These methods specifically work by taking into account two photos of the same object in various poses that were obtained by artificial deformations or from video sequences. These images are referred to as the source and target images. However, the network needs to have a tight information bottleneck for these approaches to function and successfully extract the semantic representation. If this requirement is not met, the appearance information will contaminate the semantic representation, and the disentanglement will be subpar. This is what we refer to as leaking. Most self-supervised algorithms [9] are inherently leaky because of this.This is what we refer to as leaking. Due to leaking, the majority of self-supervised algorithms [9] are intrinsically constrained to concentrate on keypoint-based low-dimensional object representation. While segmentation maps in our situation serve as the semantic representation bottleneck.

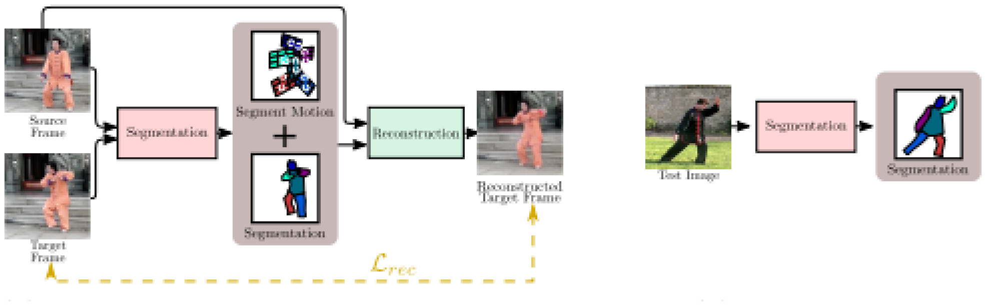


Fig 1 Co-part segmentation via a self-supervised approach

This study addresses the issue and presents a brand-new deep architecture that predicts accurate semantic representations without leakage. With the use of a per-segment motion representation and the predicted segments from the target frame, our method reconstructs the desired image (Fig. 1). We specifically suggest a part-based network, which deforms every component from the source picture to match its corresponding component in the target image. The projected motion representation for each part serves as the basis for this distortion. To further improve background-foreground separation, we also provide a background visibility mask.Two datasets, Tai-Chi-HD [10] and VoxCeleb [11], were used to evaluate our methodology.

We illustrate the efficacy of our strategy and the justification for our architectural decisions by an extensive comparison with prior approaches on co-part segmentation [8] and an in-depth ablation study.

In conclusion, we have provided two contributions. By outlining the issue of video-based co-part segmentation, we demonstrate how motion information may and ought to be used to infer useful object parts. We anticipate that this new research direction will be furthered by our efforts. We also provide a brand-new deep architecture for co-part segmentation. By demonstrating that complicated segmentation maps in addition to object landmarks may be inferred by disentanglement inside a reconstruction framework, our method enhances the state-of-the-art for self-supervised object parts finding.

.

CHAPTER 2
LITERATURE SURVEY

A specific Convolutional Neural Network model was utilised to compare the produced face areas with their surrounding regions in order to find artefacts for the Detection of Deep Fakes Face Warping Artefacts [12]. There were two face artefacts in this piece. Their approach is based on the fact that the existing deepface algorithm can only produce pictures of a certain resolution, which then requires further transformation to match the faces that need to be replaced in the original video. The temporal analysis of the frames was not taken into account in their methodology. In Detection by Eye Blinking [13], a novel technique for identifying deep fakes is described. Eye blinking is a key feature that determines whether a video is deep fake or authentic.The clipped frames of eye blinking were temporally analysed using the Long-term Recurrent Convolutional Network (LRCN). Since today's deep fake creation algorithms are so advanced, the absence of eye blinking cannot be the only indicator of a deep fake. Other factors, such as dental enchantment, facial wrinkles, incorrect brow positioning, etc., must be taken into account in order to recognise profound fakes. Capsule networks are used to detect fake pictures and videos in a variety of situations, such as replay attack detection and computer-generated video detection. This technique is described in Capsule networks to detect fake images and videos [14]. They employed random noise in the training phase of their approach, which is a poor choice. Even so, the model showed promise in their dataset, but because of training-stage noise, it might not work well on real-time data. On noiseless and real-time datasets, our approach is suggested to be trained. Recurrent Neural Network (RNN) [15] for deepfake detection employed the strategy of employing RNN for sequential frame processing in addition to the pre-trained ImageNet model. They used the HOHO [16] dataset, which only included 600 videos. Their dataset only includes a few of the same kinds of films, which could not work well with real-time data. We will use a sizable quantity of Real Time data to train our model.

In pristine and deep false portrait video pairings, the Synthetic Portrait Videos utilising Biological Signals [17] technique extracts biological signals from face areas. used modifications to extract the signal properties from feature vector and photoplethysmography (PPG) maps, compute the spatial coherence and temporal consistency, and then train a probabilistic Support Vector Machine (SVM) and a Convolutional Neural Network (CNN). The classification of the video as a deep fake or immaculate is then determined using the average of authenticity probability. Independent of the generator, content, resolution, and video quality, Fraudulent Catcher accurately identifies fake material. Formulating a differentiable loss function that follows the suggested signal processing stages is not a simple operation since lack of a discriminator results in the loss in their discoveries to retain biological signals.

Now that the landmarks must be learned for the development of deep fakes, a number of self-supervised learning techniques have been suggested. Thewlis et al.'s [18] proposal to develop a keypoint detector based on geometric priors compels the network to be equivalent to affine and thin spline transformations due to their training loss. This method was expanded by Zhang et al. [19] utilising an auto-encoder architecture, where the bottleneck data is the coordinates of the landmarks. [14] use a similar auto-encoding technique.Siarohin et al. [20] suggested animation techniques for video creation that find landmarks made particularly to convey motion data. Our approach is based on the motion information, just like Siarohin et al. [20]. However, even during the inference, animation algorithms take at least two frames to forecast keypoint neighbourhoods, even assuming a comparable part-based motion model.As a result, Siarohin et al[20] .'s predictions are heavily reliant on the other frame of a pair. For instance, these approaches won't forecast any neighbourhoods if there isn't enough motion between two frames, and (ii) picture animation methods may incorporate a lot of background motion into the motion prediction. By producing distinct frame-based predictions, our method, in contrast, encodes more semantically significant portions.

The majority of co-part segmentation techniques work under supervision [7]. Here, we limit ourselves to reviewing earlier research on unsupervised co-part segmentation. To identify object component information, recent techniques suggest studying the internal feature representations produced from pre-trained ConvNets [21]. By using Non-negative Matrix Factorization (NMF) on features calculated from a pre-trained ConvNet, Collins et al. [22] proposed a method to estimate segments corresponding to object sections. This excludes geometric priors and necessitates an expensive optimization step at the time of inference. In order to extract object pieces from unlabeled films and their associated hierarchical structure and dynamical model, Xu et al. [24] suggested a deep model. They do, however, presuppose that precomputed motion data is accessible.Our technique calculates the motion within the same design in a different way. A self-supervised deep learning method for co-part segmentation from static pictures was recently proposed by Hung et al. [12]. To enforce geometric, equivariance, and semantic consistency requirements, a network is trained employing a variety of losses. Hung et al .'s [25] analysis is based on a sizable collection of identical item category-related unlabelled photos. For co-part segmentation in this study, we also use a self-supervised learning approach. But rather than taking into account static pictures, we suggest training our model on a set of unlabeled movies. Our goal is to acquire segments that correspond to clusters of pixels connected with object pieces moving collectively by utilising motion information.

As a last observation, we want to make clear that copart segmentation is distinct from co-image and co-video segmentation [26]. The goal of co-image and co-video segmentation is to find pixels inside photos and videos that correspond to common foreground objects of the same class. The ultimate objective in both situations is to segment a single foreground object, not to find object components.

CHAPTER-3

PROPOSED METHODOLOGY

3.1 DETECTION OF DEEPFAKE

3.1.1 Data-set Gathering

To improve the model's ability to make predictions in real time. We acquired the information from many publicly accessible data-sets, including Celeb-DF[1]. To further improve accuracy and speed of real-time detection on various types of movies, we combined the datasets we had previously gathered. We evaluated a set of films that were 50% real and 50% phoney in order to prevent the model's training bias.

3.1.2 Pre-processing

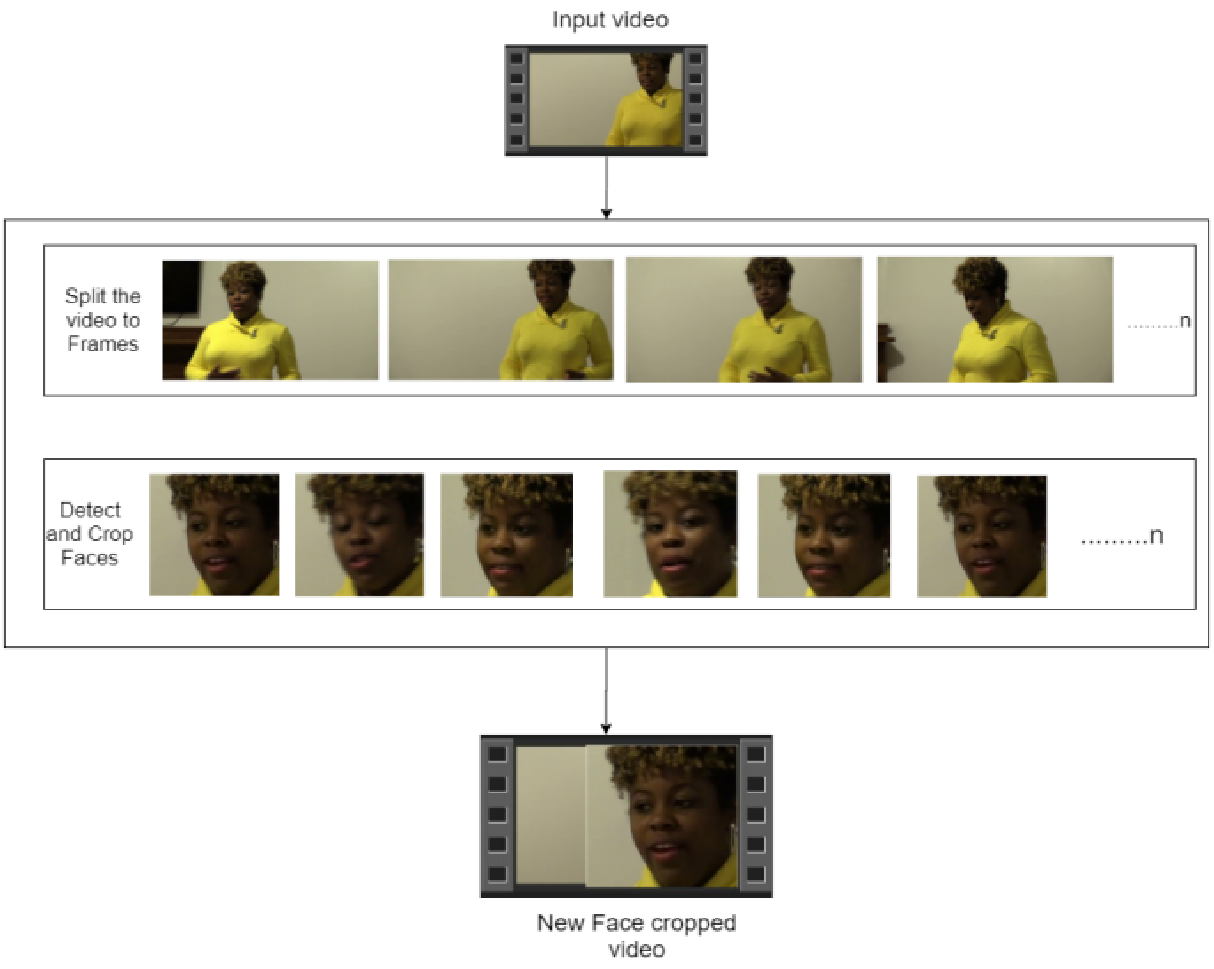


Figure: 3. 1 Preprocessing

All unnecessary noise and preprocessing are taken out of the videos. Only the necessary section of the video, which is the face, is recognised and trimmed. Splitting the video into frames is the first stage in the preparation of the video. The face is identified in each frame of the movie after it has been divided into frames, and the frame is then cropped along the face. Later, the video's individual frames are combined to create a new video from the chopped frame. Each film in the process is done thus in order to create a processed dataset of face-only movies. During preprocessing, the frame that doesn't have a face is disregarded.

We chose a threshold value based on the mean of the total frames count of each movie in order to retain the consistency of the number of frames. Limited computing power is another factor to consider when choosing a threshold number. As a movie of 10 seconds at 30 frames per second (fps) will include a total of 300 frames, processing all 300 frames at once in the experimental scenario is quite computationally challenging. Therefore, we chose 150 frames as the threshold number based on the computing capabilities of our Graphic Processing Unit (GPU) in an experimental setting. We have only preserved the first 150 frames of the movie to the new video while saving the frames to the new dataset.

We have taken into consideration the first 150 frames sequentially rather than randomly in order to show how Long Short-Term Memory (LSTM) should be used. The freshly produced video is stored with a resolution of 112 x 112 and a frame rate of 30 fps.

3.1.3 Data set split

The dataset is divided into a train dataset and a test dataset with a ratio of 70% train movies (4,200) and 30% test videos (1,800). The train and test split is evenly divided, with 50% genuine videos and 50% bogus videos in each split.

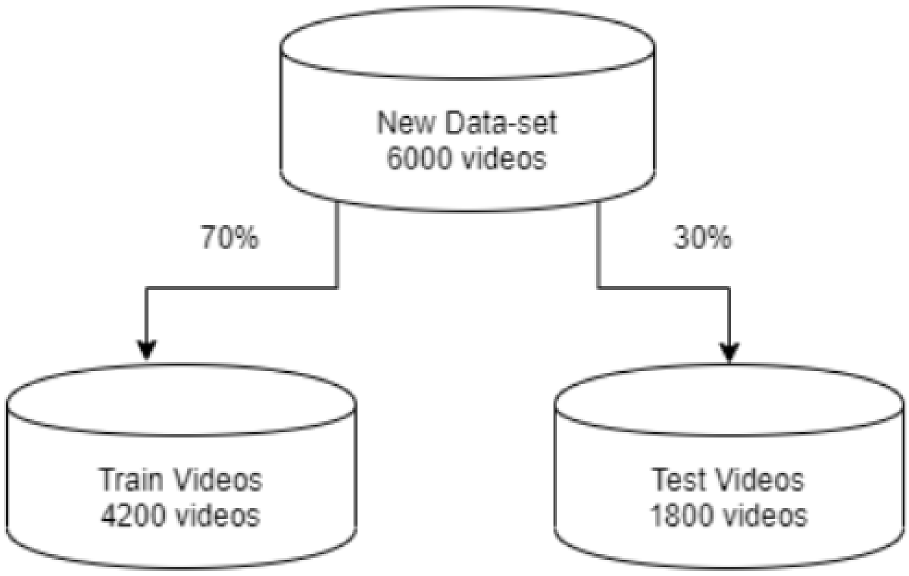


Figure 3.2 Train test split

3.1.4 Model architecture

Our model is a CNN and RNN hybrid. A LSTM network is trained to categorise the video as deep fake or pristine based on the attributes that were retrieved at the frame level using the pre-trained ResNext CNN model. The labels of the movies are loaded and fitted into the model for training using the Data Loader on the training split of the videos.

ResNext :

We used ResNext's pre-trained model for feature extraction rather than building the algorithm from scratch. ResNext is a Residual CNN network that has been enhanced for superior functionality on deeper neural networks. We utilised the resnet50 32x4d model for the experimental purpose. A ResNext with 50 layers and 32 x 4 dimensions has been utilised.

The network will then be fine-tuned by adding any additional necessary layers and choosing an appropriate learning rate to correctly converge the gradient descent of the model. The sequential LSTM input is the 2048-dimensional feature vector from ResNext's last pooling layers.

LSTM for Sequence Processing:

The input to the LSTM is fitted with 2048-dimensional feature vectors. In order to accomplish our goal, we are utilising a single LSTM layer with 2048 latent dimensions, 2048 hidden layers, and a 0.4 likelihood of dropout.

The frames are processed sequentially using LSTM in order to do a temporal analysis of the video by comparing the frame at second t with the frame at second t-n. In where n is the number of frames before it. Leaky Relu activation function is another component of the concept.The model can learn the average rate of correlation between the input and output using a linear layer with 2048 input features and 2 output features. The model employs an adaptive average pooling layer with output parameter 1.

This results in a picture with the desired output dimensions of H x W. A Sequential Layer is utilised for the sequential processing of the frames.

The batch training is carried out in groups of four. To determine the model's level of confidence during prediction, a SoftMax layer is employed.

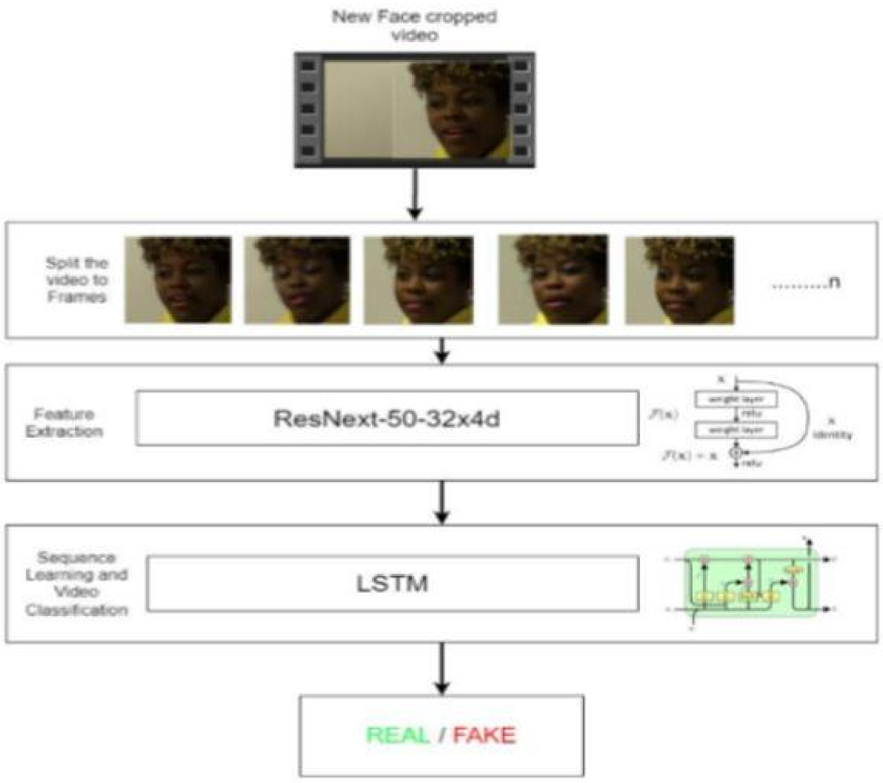


Figure 3.3 Overview of our model

3.1.5 Hyper-Parameter tuning

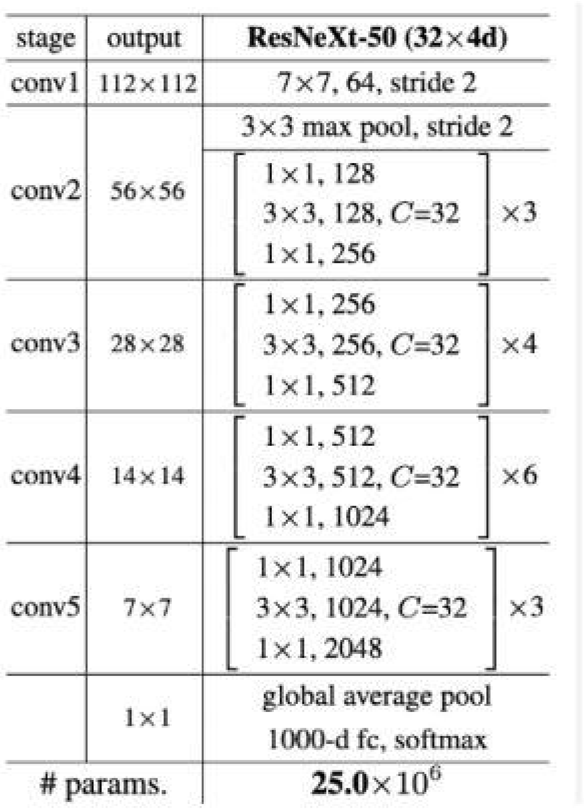
It involves selecting the ideal hyper-parameters to get the highest level of accuracy. after several iterations of the model. We select the ideal hyper-parameters for our dataset. Adam[21] optimizer with the model parameters is utilised to enable the adjustable learning rate. To attain a better global minimum of gradient descent, the learning rate is set to 1e-5 (0.00001). 1e-3 is the chosen weight decay. Since this is a classification issue, the cross-entropy technique is employed to determine the loss. Batch training is utilised to effectively utilise the computing power at hand. We consider the batch size to be 4. In our development environment, a batch size of four has shown to be the best amount for training.The Django framework is used to create the application's user interface. Django is utilised to make the application scalable in the future. A tab for browsing and uploading videos may be found on the user interface's index.html page. The model is then given the uploaded video, and it makes a prediction. Whether the video is authentic or not, along with the model's level of confidence, is returned by the model. On the screen of the currently playing video, the output is presented in the predict.html file.

3.1.6 Model details

The model consists of following layers:

• **ResNext CNN** :

The Residual Convolution Neural Network's pre-trained model is applied. Resnet50 32x4d()[27] is the model's name. 50 layers and 32 x 4 dimensions make up this model. The model's thorough implementation is shown in Figure.



stage	output	**ResNeXt-50 (32×4d)**
conv1	112×112	7×7, 64, stride 2
conv2	56×56	3×3 max pool, stride 2
		$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128, C{=}32 \\ 1\times1, 256 \end{bmatrix} \times 3$
conv3	28×28	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256, C{=}32 \\ 1\times1, 512 \end{bmatrix} \times 4$
conv4	14×14	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512, C{=}32 \\ 1\times1, 1024 \end{bmatrix} \times 6$
conv5	7×7	$\begin{bmatrix} 1\times1, 1024 \\ 3\times3, 1024, C{=}32 \\ 1\times1, 2048 \end{bmatrix} \times 3$
	1×1	global average pool 1000-d fc, softmax
# params.		**25.0**×10^6

Table 3.1 ResNext Architecture

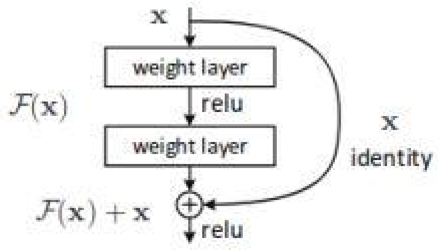

x
weight layer
$\mathcal{F}(\mathbf{x})$
relu
weight layer
x
identity
$\mathcal{F}(\mathbf{x}) + \mathbf{x}$
relu

Figure 3.4 ResNext Working

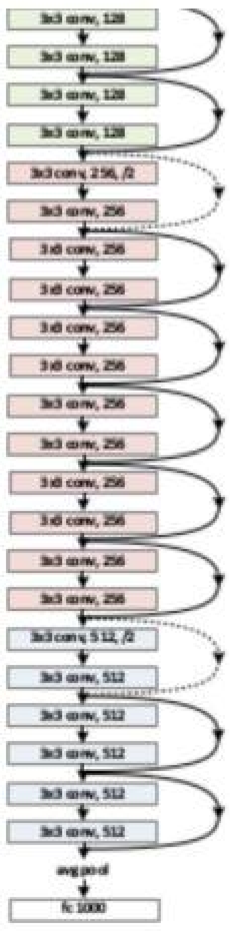


Figure 3.5 Overview of ResNext Architecture

• Sequential Layer :

Sequential is a collection of Modules that may be performed concurrently and layered on top of one another. The feature vector provided by the ResNext model is stored in an ordered manner using a sequential layer. so that it may be sent consecutively to the LSTM.

• LSTM Layer :

For analysing sequences and identifying the temporal shift between the frames, LSTM is employed. The input to the LSTM is fitted with 2048-dimensional feature vectors. In order to accomplish our goal, we are utilising a single LSTM layer with 2048 latent dimensions, 2048 hidden layers, and a 0.4 likelihood of dropout. The frames are processed sequentially using LSTM in order to do a temporal analysis of the video by comparing the frame at second it with the frame at second t-n. In which n is the number of frames before it.

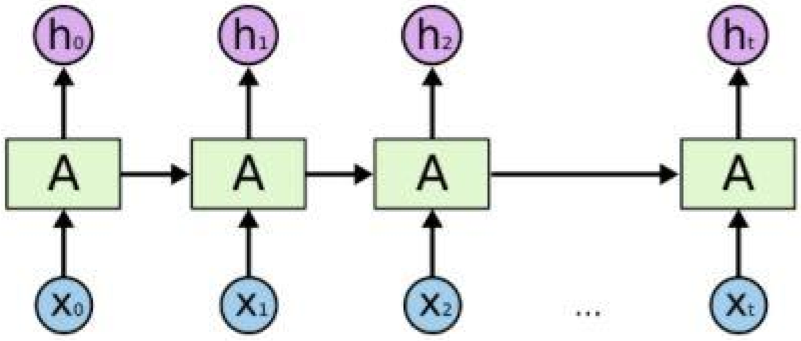


Figure 3.6 Overview of LSTM Architecture

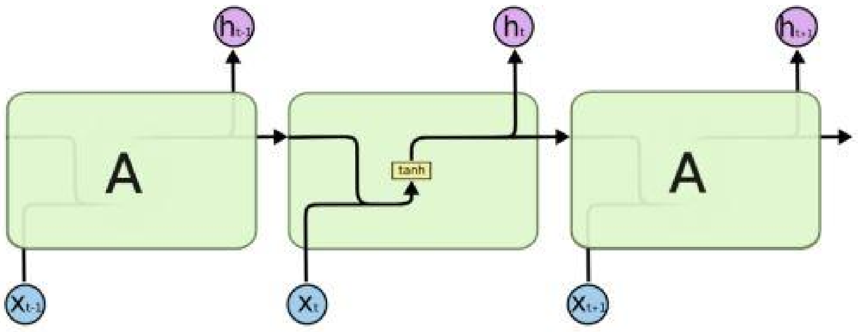


Figure 3.7 Internal LSTM Architecture

• ReLU:

A Rectified Linear Unit is an activation function with a raw output in all other cases and an output of 0 if the input is less than 0. In other words, the output is identical to the input if the input is higher than 0. ReLU's functioning is more similar to how organic neurons function.

ReLU is non-linear and has the benefit of not having any backpropagation mistakes, unlike the sigmoid function. It is also relatively quick to develop models based on ReLU for bigger Neural Networks.

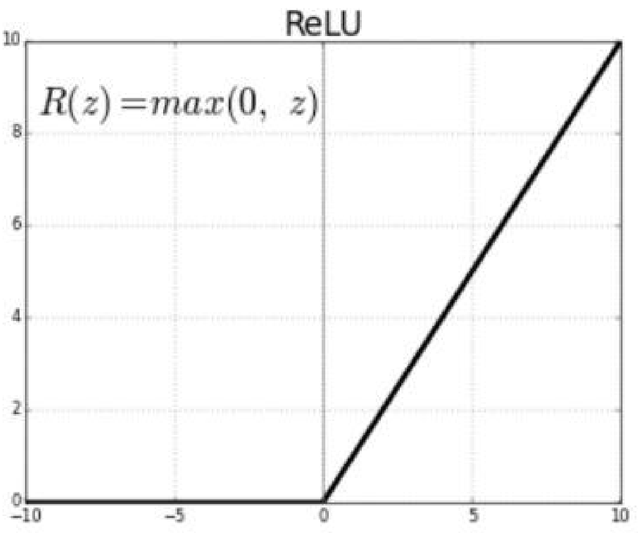


Figure 3.8 ReLU activation function

• Dropout Layer :

By randomly changing the output for a specific neuron to 0, a dropout layer with a value of 0.4 is employed to prevent overfitting in the model and can aid in model generalisation. When the output is set to 0, the cost function becomes more susceptible to nearby neurons, which alters how the weights will be changed throughout the backpropagation process.

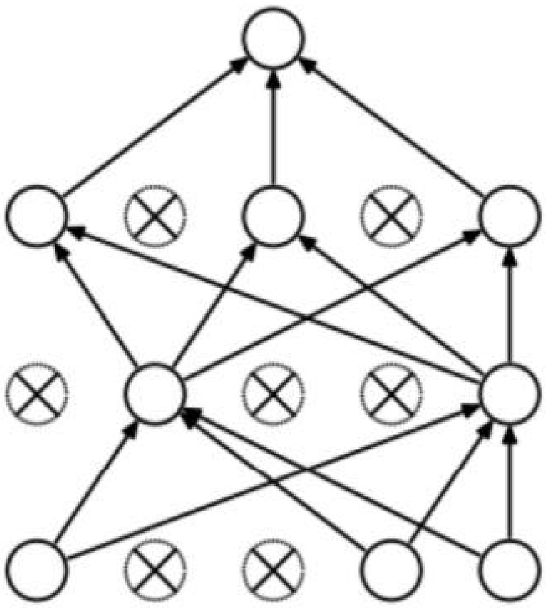


Figure 3.9 Dropout layer overview

• Adaptive Average Pooling Layer :

It is used to extract low level information from the neighbourhood and minimise variance and computation complexity. The model makes use of a two-dimensional adaptive average pooling layer.

3.1.7 Model Training Details

• Train Test Split: The dataset is divided into a train dataset and a test dataset, with 4,200 train films making up 70% of the dataset and 1,800 test videos making up the remaining 30%. The train and test split is evenly divided, with 50% genuine videos and 50% bogus videos in each split. Please see figure 7.6

• Data Loader: With a batch size of 4, it loads the movies and their labels.

• Training: Using the Adam optimizer, training is carried out for 20 epochs with a learning rate of 1e-5 (0.00001) and a weight decay of 1e-3 (0.001).

• Adam optimizer[21]: Adam optimizer with the model parameters is used to enable the adjustable learning rate.

• Cross Entropy: In order to compute the loss function Because we are training a classification problem, the cross entropy technique is applied.

• Softmax Layer: A kind of squashing function is a softmax function. Functions that are suppressed have an output range of 0 to 1. As a result, the result may be immediately understood as a likelihood. The probability of many classes may be determined simultaneously using softmax functions, which are multi-class sigmoids. A softmax layer is often the last layer employed in neural network functions since the outputs of a softmax function may be understood as a probability (i.e., they must add to 1). It's crucial to remember that a softmax layer needs to contain exactly as many nodes as the output layer. The output nodes in our scenario are two: REAL or FAKE. Additionally, Softmax layer gives us the confidence (probability) of prediction.

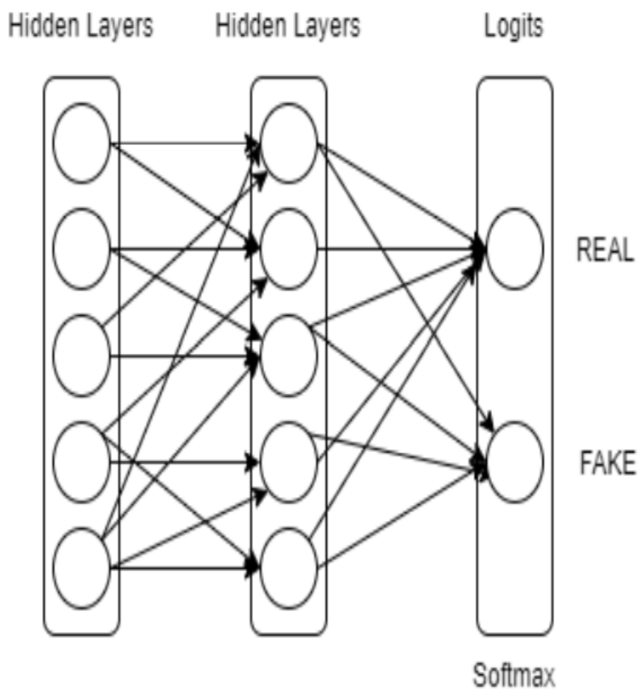


Fig 3.10 Softmax Layer

• Confusion Matrix:A classification problem's predicted outcomes are compiled in a confusion matrix. Count values are used to summarise the number of accurate and inaccurate predictions for each class. The confusion matrix's secret lies in this. The classification model's predictability confusions are displayed in the confusion matrix. It offers us knowledge about the sorts of errors that are being produced, which is more essential than just the errors that a classifier is making. Our model is assessed, and the accuracy is computed, using a confusion matrix.

• Export Model: We have exported the model once it has been trained. in order for real-time data predictions to be made using it.

• The model is loaded in the application

• Preprocessing is done on the fresh video before it is sent to the loaded model for prediction

•The trained model makes the forecast and reports whether the video is authentic or false, along with the prediction's level of confidence.

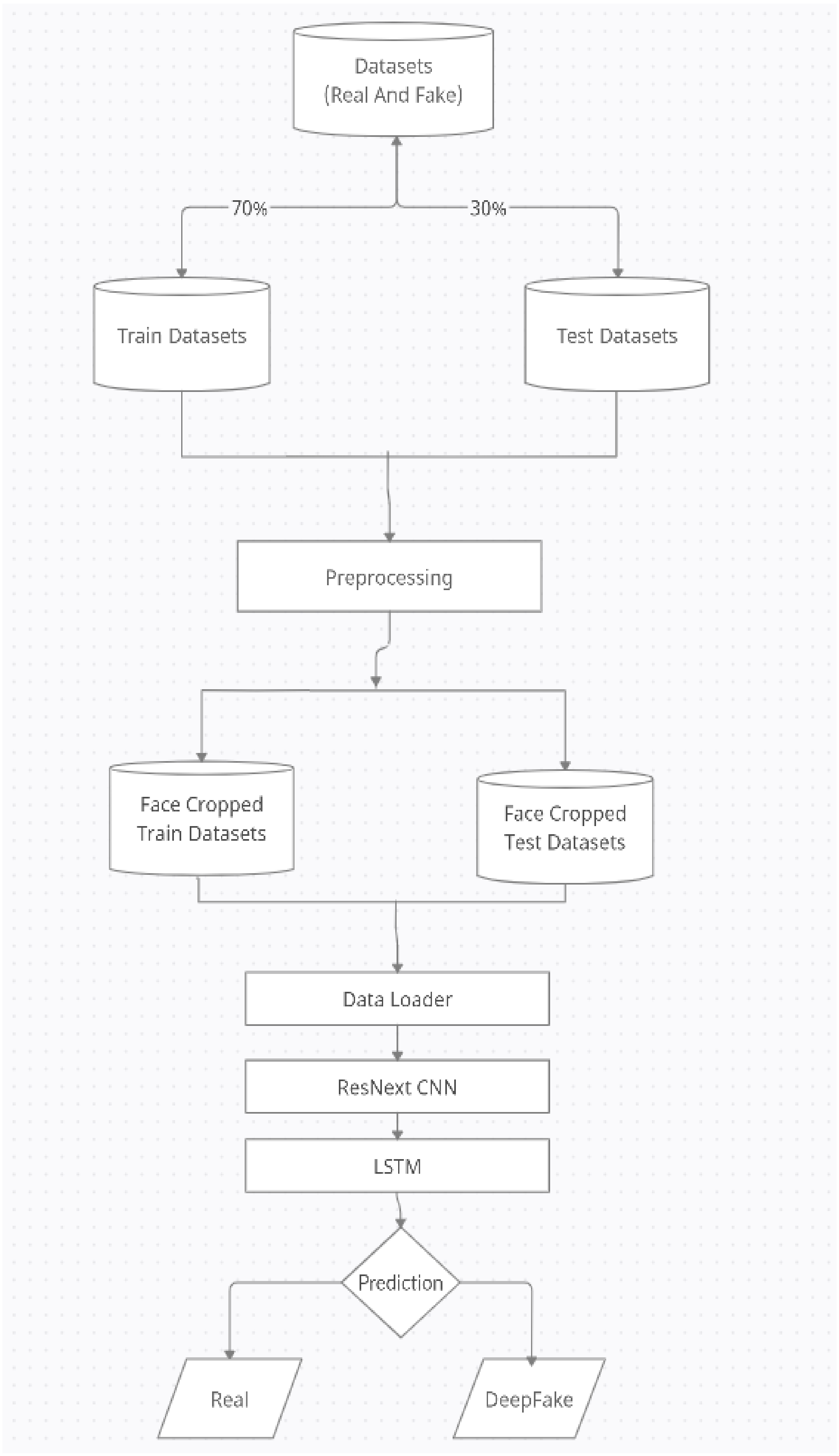


Fig 3.11 Training Flow of the model

3.1.8 Model Prediction Details

• The model is loaded in the application

• The new video for prediction is preprocessed and passed to the loaded model for prediction

• The trained model makes the forecast and reports whether the video is authentic or false, along with the prediction's level of confidence.

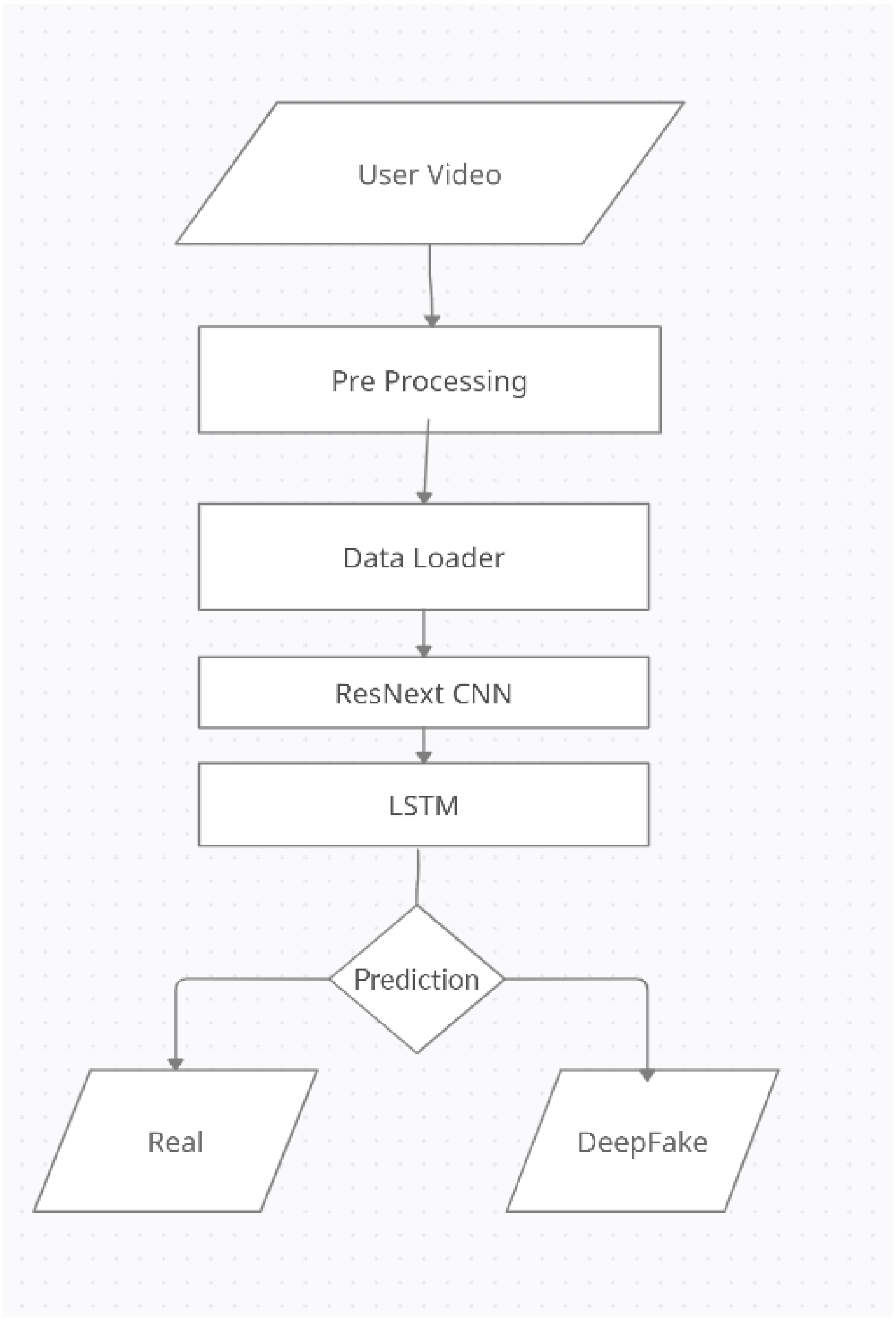


Fig 3.12 Prediction Flow

3.2 GENERATION OF DEEPFAKE

In this study, our focus is on training a deep neural network that, at the moment of inference, accepts a single picture as input and produces co-part segments. We presume that there is access to a sizable library of films with objects belonging to the same category at the time of training. The proposed model is trained using pairs of identical video frames with the spatial dimensions $X_S \in \mathrm{R}^{3 \times H \times W}$ and $X_T \in \mathrm{R}^{3 \times H \times W}$, which are referred to as the source and target frames, respectively. Keep in mind that the two frames were chosen at random from the whole video sequence. There are two key modules that make up our framework. Segmentation Module, the first module, is in charge of removing from the input frames the semantic segments and the motion associated with those segments. In order to separately divide the frames X_S and X_T into K+1 parts—K segments for the foreground and one for the background—we use a neural network.

We also want to train a generator network in our self-supervised environment, which reconstructs the target frame given the source frame and the target segmentation. Reconstructing the target frame using its segmentation maps instead of anticipating the intended semantic segmentation often results in pathological solutions that employ Y_T to encode X_T. Due to the very large dimension of Y_T (usually 6464), segmentation looks to have a particularly severe form of this difficulty compared to unsupervised landmark identification. In this instance, Y_T is too big to act as an information bottleneck, enabling the target frame's details to be revealed to the generator. As a result, we indirectly recreate the target frame using the predicted segmentation. To put it more explicitly, we suggest that the source and target frames' motion be modelled, together with the segmentation Y_T, in order to rebuild the target frame. In order to do this, we present the segment motion, which explains how each segment's source and destination frames move in relation to one another.

3.2.1 Segmentation Module

We describe how we simulate the motion of each segment in detail in this section. With the goal of getting segments that combine pixels corresponding to moving object sections together, we create our model. An affine transformation is used to simulate the mobility of the pixels inside each segment. In other words, an affine transformation on the support corresponding to each segmented section may be used to approximate the optical flow F. In order for our affine model to be valid, the encoder network should deliver segments. Otherwise, it would result in an incorrect optical flow F and a substantial loss during reconstruction.Let $Y_T^k \in [0,1]^{H' \times W'}$ represent the k th channel of the segmentation Y_T in formal terms. For the purposes of nomenclature, we will assume that Y_T and Y_S are binary tensors even though they have continuous values. As stated in $Y_T^k = \{z \in [1,H^0] \times [1,W^0] \mid Y^k_T[z] = 1\}$, we define Y_T^k as the collection of locations related to the segment k. We also present Y_S^k for source segmentation in a similar manner.

3.2.2 FLOW MODEL

We now go into more depth about how the segment motion representations and the expected segmentation are combined to produce optical flow F. We use the model to get K optical flow fields $F^k \in \mathbb{R}^{2 \times H0 \times W0}$ of F in each individual segment Y_F^k for each segment. The partial optical flow for the pixels of the segment Y_T^k between the source and target frames can be represented by the tensor F^k. Keep in mind that the backdrop is linked to the latest optical flow field, F^{k+1}. As a result, we put F^{k+1} = z and assume that the backdrop is unchanging. We assume that each component of the object corresponds to a segment in Y_T, and that its motion may be represented by the segment motion descriptor to which it relates.

Y_T^k are continuous tensors in reality. Eq. (3) may be understood as a soft assignment model that chooses a motion vector for each pixel point as a result. Gradient-based optimization is made easier by the usage of continuous tensors.

Background visibility mask - The backdrop visibility mask $V \in [0,1]^{H0 \times W0}$ is also estimated. It shows which background pixels in the source frame are hidden by the foreground item. Later, the Reconstruction Module will suppress data pertaining to the obscured portions of the backdrop. Superior background/foreground separation is enforced by the use of the background visibility mask. Bad backdrop visibility maps will result from poor segmentation maps, which will lower the quality of the final reconstruction. The source and target segmentations are used to calculate V .By combining the K object sections of $O^S = \sum_{k=1}^{K} Y_S^k$, we are able to predict the positions of the foreground objects in the source frame. Then, occluded locations are those that correspond to the foreground in the source frame but the background in the target (Y^{K+1}_T=1). The background visibility mask's formal formula is:

$$V = = 1 - (\boldsymbol{Y}_T^{K+1} \otimes \boldsymbol{O}_S \quad (1)$$

The background visibility, which we will refer to as false fg later, is an essential regularisation for minimising areas of the background that are wrongly categorised as the foreground. In fact, a considerable quantity of information that would be helpful for background reconstruction will be hidden if the source frame had a big amount of misleading fg. Optimization of the reconstruction error will therefore result in reduction of *false fg*.

Our backdrop visibility does not enhance reconstruction in comparison to the occlusion map of [10], which identifies the areas that must be painted in order to increase reconstruction quality. In fact, it even does it harm. Since information about the target frame will leak through Y_T^{K+1} It is important to highlight that giving the generator background visibility might be difficult.

3.2.3 RECONSTRUCTION AND TRAINING

In this part, we'll go through how we recreate the target frame using the predicted optical flow. Then, we go into depth about how we do our training. By bending the source frame characteristics in accordance with the predicted optical flow, we use a generator network G to reconstruct the target frame. Several current pose-guided generation frameworks [44,33,45,31,13] have adopted a similar method. G features a deformation layer in its bottleneck and an encoder-decoder architecture. The estimated optical flow F, the feature map R C, the background visibility mask V, and the deformation layer's input are all required. In order to hide feature map locations that are hidden on the segmentation maps, we employ V.

As a result, we lessen the influence of the characteristics found in the hidden areas. Keep in mind that the source segmentation will be penalised if the features in that area were relevant for reconstruction (this occurs in the case of *false fg*).

Training Losses We train our model from beginning to end. Our primary motivating loss is the loss from rebuilding. It is based on the Johnson et al. [29] perceptual loss and measures reconstruction quality using a pre-trained VGG-19 [30] network.

The reconstruction loss is defined as follows given the input target frame X_T and the accompanying rebuilt frame $\bar{X}_T$

$$\mathcal{L}_{rec}(\bar{\boldsymbol{X}}_T, \boldsymbol{X}_T) = \sum_{i=1}^{I} \left| \Phi_i(\bar{\boldsymbol{X}}_T) - \Phi_i(\boldsymbol{X}_T) \right| \tag{2}$$

where Φi(·) represents the i-th channel feature that was extracted from a particular VGG-19 [34] layer, and I stands for the layer's number of feature channels. Similar to MS-SSIM [32], this loss is also calculated at various resolutions.

To further drive the network to predict segment motion representations that are consistent with regard to known geometric transformations, we use the equivariance loss in a manner similar to other unsupervised keypoint identification approaches [33].

We apply the equivariance loss (also known as L_{eq}) with regard to affine transformations defined by p^k and A^k, much like in [10]. Last but not least, L_{rec} and L_{eq} are added to create the total loss function. We begin with the pre-trained weights of [10] to aid convergence and enhance performance, but we think any unsupervised keypoint detection approach [34] may be applied. We also tried out starting from fresh when training. We found that the model does pick up on relevant portions, although their accuracy isn't the best.

CHAPTER 4
RESULTS AND DISCUSSIONS

4.1 Detection Model Results

The model's output will include the model's confidence level and a determination of whether the video is authentic or a deep fake. The figure below shows one instance.

Our approach uses ResNext CNN for frame level detection and RNN and LSTM for video classification. We achieved an accuracy of almost 90% using the Celeb-DF datasets.

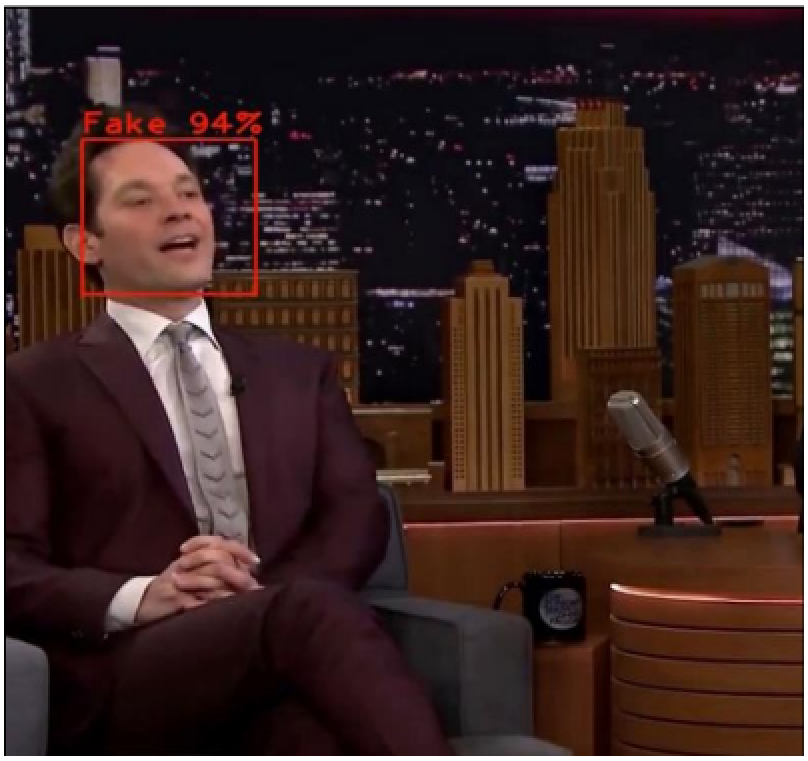


Fig 4.1 Detection Result

4.2 Generation Model Results

Our methodology generates semantic portions that are more consistent across various objects and in which the primary object and background are distinct from one another.

In further detail, the results of the video editing are produced frame-by-frame, and our segmentation module predicts the semantic masks for both the target frame and the source picture, together with the flow F for each mask. Then, by deforming the source features, this part-based F is utilised to align the source features with the matching target features. The target frame's lower jaw features are then removed using the mask of interest to be swapped (for example, the lower jaw), and the necessary features from the source picture are used to fill the space. Our reconstruction module then decodes the resultant intermediate characteristics to produce the final image, in which the lower jaw seems to have been switched from the original image.

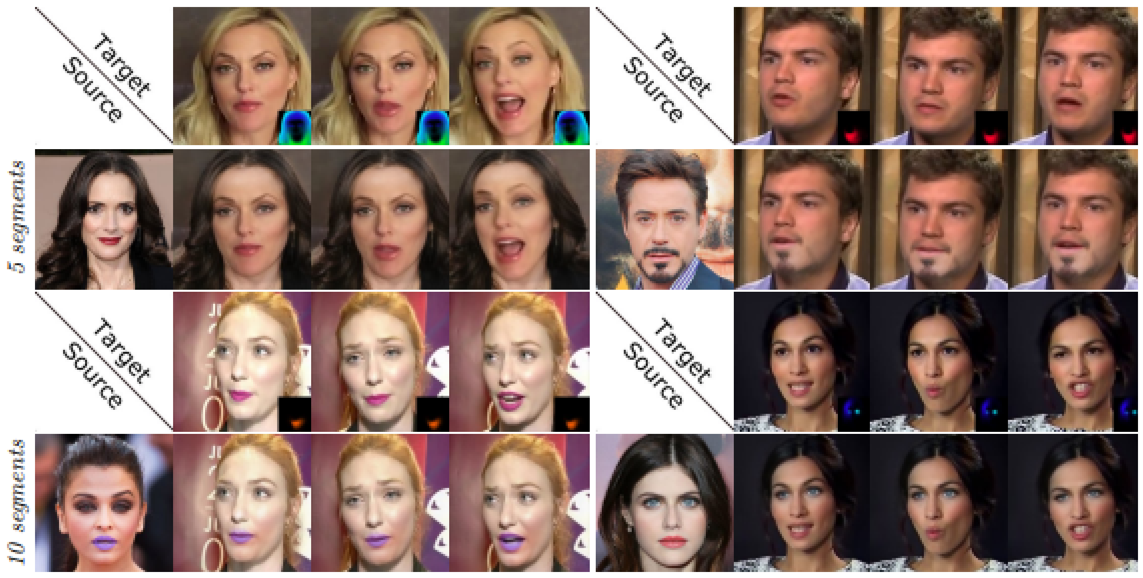


Figure 4.2 Generation results

We experimented with two models that are trained to predict semantic portions with K=5 and K=10, respectively, because the outcomes of video editing vary depending on the application of interest.. We use K=5 when we want to change large facial parts such as blonde hair → black hair and clean shave → goatee beard.

On the other hand, we accomplish a fine-grained editing by using a model with K=10 where we alter only small parts of the face, as in red lips → purple lips and black eyes → turquoise eyes .To produce a smoother transition between areas and prevent a sudden shift in skin tone, we employed continuous masks for video editing applications.

CHAPTER 5
CONCLUSION

5.1 CONCLUSION

Along with the confidence in the suggested model, we offered a neural network-based method for classifying the video as deep fake or real. Our technique can accurately anticipate the result after analysing 1 second of video (10 frames per second). In order to detect changes between the t and t-1 frames, we developed the model using a pre-trained ResNext CNN model to extract the frame-level features and LSTM for temporal sequence processing. Our model is capable of processing videos with 10, 20, and 40 frames.
Any built system may always be improved, especially when it was created utilising the most recent technology and has a promising future.
•For user convenience, web-based platforms can be upgraded to a browser plugin.
• The system can be improved to detect complete body deep fakes in addition to only face deep fakes, which is what it can now only detect.

Regarding the generation of deep fakes, we introduced a unique self-supervised method for co-part segmentation that makes use of motion data automatically derived from video streams to forecast superior segmentation maps from still photos. We described how the leakage problems are the key obstacle to creating a self-supervised reconstruction-based segmentation model, and we then developed a unique part-based motion formulation that makes it possible to solve this issue. In order to improve the separation of the item from the backdrop, we also add a background visibility mask to our model. We illustrate the validity and superiority of our strategy through a thorough experimental examination and comparisons with cutting-edge techniques.

References

[1] Yuezun Li , Xin Yang , Pu Sun , Honggang Qi and Siwei Lyu “Celeb-DF: A Large-scale Challenging Dataset for DeepFake Forensics” in arXiv:1909.12962

[2] Face app: https://www.faceapp.com/ (Accessed on 26 March, 2020)

[3] Face Swap : https://faceswaponline.com/ (Accessed on 26 March, 2020)

[4] Yu, R., Wang, X., Xie, X.: Vtnfp: An image-based virtual try-on network with body and clothing feature preservation. In: CVPR (2019)

[5] Dong, H., Liang, X., Gong, K., Lai, H., Zhu, J., Yin, J.: Soft-gated warping-gan for pose-guided person image synthesis. In: NIPS (2018)

[6] Nirkin, Y., Keller, Y., Hassner, T.: Fsgan: Subject agnostic face swapping and reenactment. In: CVPR (2019)

[7] Chen, X., Mottaghi, R., Liu, X., Fidler, S., Urtasun, R., Yuille, A.: Detect what you can: Detecting and representing objects using holistic models and body parts. In: CVPR (2014)

[8] Collins, E., Achanta, R., Susstrunk, S.: Deep feature factorization for concept discovery. In: ECCV(2018)

[9] Jakab, T., Gupta, A., Bilen, H., Vedaldi, A.: Unsupervised learning of object landmarks through conditional image generation. In: NIPS (2018)

[10] Siarohin, A., Lathuili`ere, S., Tulyakov, S., Ricci, E., Sebe, N.: First order motion model for image animation. In: NeurIPS (2019)

[11] Nagrani, A., Chung, J.S., Zisserman, A.: Voxceleb: a large-scale speaker identification dataset. In: INTERSPEECH (2017)

[12] Yuezun Li, Siwei Lyu, “ExposingDF Videos By Detecting Face Warping Artifacts,” in arXiv:1811.00656v3.

[13] Yuezun Li, Ming-Ching Chang and Siwei Lyu “Exposing AI Created Fake Videos by Detecting Eye Blinking” in arXiv:1806.02877v2.

[14] Huy H. Nguyen , Junichi Yamagishi, and Isao Echizen “ Using capsule networks to detect forged images and videos ” in arXiv:1810.11215.

[15] 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, pp. 1-6.

[16] I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld. Learning realistic human actions from movies. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 1–8, June 2008. Anchorage, AK

[17] Umur Aybars Ciftci, ˙Ilke Demir, Lijun Yin “Detection of Synthetic Portrait Videos using Biological Signals” in arXiv:1901.02212v2

[18] Thewlis, J., Bilen, H., Vedaldi, A.: Unsupervised learning of object landmarks by factorized spatial embeddings. In: ICCV (2017)

[19] Zhang, Y., Guo, Y., Jin, Y., Luo, Y., He, Z., Lee, H.: Unsupervised discovery of object

landmarks as structural representations. In: CVPR (2018)

[20] Siarohin, A., Lathuili`ere, S., Tulyakov, S., Ricci, E., Sebe, N.: First order motion model for image animation. In: NeurIPS (2019)

[21] Bau, D., Zhou, B., Khosla, A., Oliva, A., Torralba, A.: Network dissection: Quantifying interpretability of deep visual representations. In: CVPR (2017)

[22] Collins, E., Achanta, R., Susstrunk, S.: Deep feature factorization for concept discovery. In: ECCV (2018)

[23] Sra, S., Dhillon, I.S.: Generalized nonnegative matrix approximations with bregman divergences. In: NIPS (2006)

[24] Xu, Z., Liu, Z., Sun, C., Murphy, K., Freeman, W.T., Tenenbaum, J.B., Wu, J.: Unsupervised discovery of parts, structure, and dynamics (2019)

[25] Hung, W.C., Jampani, V., Liu, S., Molchanov, P., Yang, M.H., Kautz, J.: Scops: Self-supervised co-part segmentation. In: CVPR (2019)

[26]Li, W., Jafari, O.H., Rother, C.: Deep object co-segmentation. In: Asian Conference on Computer Vision. pp. 638–653. Springer (2018)

[27] ResNext Model : https://pytorch.org/hub/pytorch_vision_resnext/ accessed on

06 April 2020

[28] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization.

arXiv:1412.6980, Dec. 2014.

[29] Johnson, J., Alahi, A., Fei-Fei, L.: Perceptual losses for real-time style transfer and super-resolution. In: ECCV (2016)

[30] Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: ICLR (2015)

[31] Zablotskaia, P., Siarohin, A., Zhao, B., Sigal, L.: Dwnet: Dense warp-based network

for pose-guided human video generation. In: BMVC (2019)

[32] Wang, Z., Simoncelli, E.P., Bovik, A.C.: Multiscale structural similarity for image

quality assessment. In: ACSSC (2003)

[33] Zhang, Y., Guo, Y., Jin, Y., Luo, Y., He, Z., Lee, H.: Unsupervised discovery of

object landmarks as structural representations. In: CVPR (2018)

[34] Siarohin, A., Lathuili`ere, S., Tulyakov, S., Ricci, E., Sebe, N.: Animating arbitrary

objects via deep motion transfer. In: CVPR (2019)