**SN Computer Science manuscript No.**
(will be inserted by the editor)

# A Machine Learning based Approach for DeepFake Detection in Social Media through Key Video Frame Extraction

**Alakananda Mitra · Saraju P. Mohanty* · Peter Corcoran · Elias Kougianos**

the date of receipt and acceptance should be inserted later

Received:01 October 2020 /Accepted:date

**Abstract** In the last few years, with the advent of deepfake videos, image forgery has become a serious threat. In a deepfake video, a person's face, emotion or speech are replaced by someone else's face, different emotion or speech, using deep learning technology. These videos are often so sophisticated that traces of manipulation are difficult to detect. They can have a heavy social, political and emotional impact on individuals, as well as on the society. Social media are the most common and serious targets as they are vulnerable platforms, susceptible to blackmailing or defaming a person. There are some existing works for detecting deepfake videos but very few attempts have been made for videos in social media.

The first step to preempt such misleading deepfake videos from social media is to detect them. Our paper presents a novel neural network-based method to detect fake videos. We applied a key video frame extraction technique to reduce the computation in detecting deepfake videos. A model, consisting of a convolutional neural network (CNN) and a classifier network, is proposed along with the algorithm. The Xception net has been chosen over two other structures - InceptionV3 and Resnet50 - for pairing with our classifier. Our model is a visual artifact-based detection technique. The feature vectors from the CNN module are used as the input of the subsequent classifier network for classifying the video. We used the FaceForensics++ and Deepfake Detection Challenge datasets to reach the best model.

Our model detects highly compressed deepfake videos in social media with a very high accuracy and lowered computational requirements. We achieved 98.5% accuracy with the FaceForensics++ dataset and 92.33% accuracy with a combined dataset of FaceForensics++ and Deepfake Detection Challenge. Any autoencoder generated video can be detected by our model.

Our method has detected almost all fake videos if they possess more than one key video frame. The accuracy reported here is for detecting fake videos when the number of key video frames is one.

The simplicity of the method will help people to check the authenticity of a video. Our work is focused, but not limited, to addressing the social and economical issues due to fake videos in social media.

In this paper, we achieve the high accuracy without training the model with an enormous amount of data. The key video frame extraction method reduces the computations significantly, as compared to existing works.

**Keywords** Deepfake · Deep Learning · Key Video Frame Extraction · Depthwise Separable Convolution · Convolution Neural Network (CNN) · Transfer Learning · Social Media · Compressed Video.

A. Mitra
Dept. of Computer Sci. and Eng., University of North Texas
E-mail: AlakanandaMitra@my.unt.edu.

S. P. Mohanty (Corresponding Author)
Dept. of Computer Sci. and Eng., University of North Texas
E-mail: saraju.mohanty@unt.edu

P. Corcoran
School of Engineering & Informatics, National University of Ireland, Galway, Ireland
E-mail: peter.corcoran@nuigalway.ie.

E. Kougianos
Dept. of Electrical Engineering, University of North Texas
E-mail: elias.kougianos@unt.edu.

## 1 Introduction

Image and video forgery are posing a threat to the society in today's world. People can artificially create any audio or video clip. Artificial intelligence, mainly machine learning, manipulates images and videos in such a way that they are often visually indistinguishable from real ones [32, 57, 58]. There are some prevalent techniques which are widely used to manipulate images/videos. Some are computer graphic based (e.g. Photoshop, GIMP, and Canva) and the rest are content changing. Deepfake, a deep learning-based method, is a serious contender among the content changing video falsification techniques. The term "deepfake" originates from the words "deep learning" and "fake". Use of deep learning networks (DNN) has made the process of creating convincing fake images and videos increasingly easier and faster. It is a technique in which a video or image of a person is manipulated by the image of another person using deep learning methods [3, 4].

In today's life, social network/media play a significant role. They can affect someone's mental health [16] and social status, though it shouldn't be that way. [7, 9]. Mobile or mini cameras let people take pictures or videos anywhere, anytime. Commercial photo editing tools [1, 5] allow anyone to create fake images/videos. So, amid multimedia forgery, we need some counter measures to protect our privacy and identity, especially in social media where a person is vulnerable [2]. In social media, when images or videos are uploaded, they get compressed and resized. So the techniques applicable to uncompressed videos might not work for highly compressed videos. In this paper, we are proposing a novel method to detect deepfake videos in compressed social media.

The rest of this paper is organized as follows: Section 2 presents the motivation of our work. Section 3 focuses on the novel contributions of this paper. Section 4 is a review of related works in this field. Our detailed work for deepfake detection is described in Section 6. Section 7 presents the theoretical perspective. Section 8 discusses experiments and results. Section 9 states the conclusions of this paper with some discussions on the directions on future works.

## 2 Deepfake is a Social and Economical Issue

Our face is our identity. People remember someone as per their face. So, when image and video forgery come into play, face manipulation becomes the most targeted one. In the last two decades, face forgery in multimedia has increased enormously. Among the reported works, an image based approach was used by Breglera et al. [12] in 1997 to generate a video. Face replacement of the actor without changing the expression was presented by Garrido et al. [22]. Real time expression transfer by Theis et al. [53] in 2015 is also important. Suwajanakorn's et al. [50] work on lip syncing to help people to understand how serious is video forgery.

Recent advances in deep learning changed the whole scenario of multimedia forgery. In 2017, a Reddit user, named Deepfake, created some fake videos using deep learning networks. Use of convolution auto encoders [52] and generative adversarial networks (GAN) [26] made the manipulated image/video so sophisticated that the synthesized videos are often visually indistinguishable from real ones. Multimedia forgery is now rampant. Today, smartphone applications to manipulate images are easily available to anybody. Some of these applications are the following: FaceApp, AgingBooth, Meitu, MSQRD, Reflect - Face Swap, and Face Swap Live.

The ability to distort reality has exceeded acceptable limits with deepfake technology. This disruptive technological change affects the truth. Many are intended to be funny, but others are not. They could be a threat to national security, democracy, and an individual's identity [14, 33]. A deepfake video can defame a person and invade their privacy [23, 43]. People have started to lose faith in the news or images/videos brought to them by the media. This can create political tension or violence. It can ruin any politician's career or a teenager's dream. The corporate world is interested in protecting their businesses from fraud or stock manipulation [54]. But deepfake has a dual nature. Advancement of deepfake video technology can be used with a positive approach. A hearing-impaired person who cannot follow a telephonic conversation, can converse in a phone or smartphone with the help of an app which assists in generating lip movements according to the audio. It can also help in making realistic multilingual movies. Unfortunately, negative uses are prevailing.

To stop catastrophic consequences by deepfake videos, Facebook, Microsoft, AWS, Partnership on AI, and some academic institutions came together to organize the Deepfake Detection Challenge and started building the Deepfake Detection Challenge (DFDC) dataset for research purposes. Google, in collaboration with Jigsaw announced another dataset FaceForensics++ (FF++) for deepfake video detection. Fig. 1 shows a deepfake video frame from Facebook.

## 3 Novel Contributions of the Current Paper

In this paper we are proposing a novel technique for detecting deepfake videos in social media using a classifier with lower computational requirements. Eventu-


**Fig. 1** Deepfakes created by Facebook to fight against a disinformation disaster - source Facebook

ally it can be applied as an edge fake video detecting tool. Our classifier network applies mainly autoencoder generated videos.

First we tried to detect each frame with our previously proposed Algorithm 1 [40]. Then, instead of detecting each and every frame, we applied our newly proposed Algorithm 2 where a key video frame extraction technique has been utilized. These two algorithms are discussed in detail in Section 7. As our detection method is based on finding changes of visual artifacts in a frame due to deepfake forgery, we assume that key frames contain all the artifacts. This simple assumption helps us to propose an algorithm with a good accuracy and much lower computational cost. Lower computational cost means that the algorithm can be deployed at the edge since the limited memory in a smart phone will not be a barrier.

The proposed network consists of two modules - (1) a convolutional neural network (CNN) for frame feature extraction, and (2) a classifier network consisting of GlobalAveragePooling2D and fully connected layers. To choose the best CNN module we followed our previous work [40]. We experimented mainly with three networks- Xception [15], InceptionV3 [51], and Resnet50 [30], and finally chose Xception network as the feature extractor. We chose only these three networks over other available CNN modules because of their smaller size. During selection of the CNN module, we used only the FF++ dataset [45]. Then we applied the newly proposed technique over the chosen CNN module with a larger dataset consisting of the FF++ and DFDC (partial) datasets. A system level overview of the network is shown in Fig. 2. Our model works on the proposed algorithm.

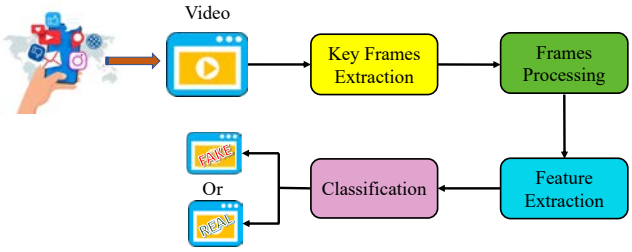



**Fig. 2** System Level Overview of the Proposed Network

### 3.1 The Problem Addressed in the Current Paper

The problem addressed in this paper lies in the very nature of how a deepfake video is created. Deepfake videos are very sophisticated and are created using different deep learning techniques - by autoencoder and generative adversarial networks (GANs). It is practically impossible for a human to distinguish between a real and a forged video when these are uploaded in social media.

Social media videos are highly compressed. Our goal is to apply our proposed technique with lower computational burden which can detect these deepfake videos at any compression level in social media, and which are generated mostly by an autoencoder. Our main goal is to lower the computations as much as possible, so that in the future we can apply the algorithm at edge devices. People can check the authenticity of videos with a limited memory device, such as a smart phone. In our previous paper [40], we tested our model only on the FF++ dataset [45]. In this paper we extended our work by testing the network with an additional dataset, DFDC [19].

### 3.2 The Challenges in Solving the Problem

In social media, uploaded images/videos are highly compressed. People check their social media account from smartphones or tablets, along with computers. The existing solutions to detect deepfake images/videos are mostly for uncompressed data and the models are not suitable for social media videos. As people check their social media account from smart phones, the model should be of smaller size too. The challenge was to solve three problems simultaneously – detecting deepfake videos, and to develop a model applicable to compressed video and also a lighter version of it. In our previous paper [40] we tried to solve the first two problems. In this paper we address the third problem too by using a computer vision technique for lower computational effort.

### 3.3 The Solution Proposed in the Current Paper

To address the above mentioned challenges we propose a novel technique of applying key video frame approach to our previously proposed neural network based model [40] which can detect deepfake videos in social media at any compression level. We limit the data by extracting only key video frames from each video. It reduces the number of frames from videos to be checked for authenticity, largely without compromising accuracy significantly. During detection, instead of checking each frame for authenticity, we check the visual artifact changes only for the key video frames. This reduces the computations. As the detection process consists of fewer data computations, this approach is one step forward to apply our network at the edge. We propose a lightweight approach compared to the highly computationally expensive existing works. Our main contributions are as follows:

- An algorithm of lower complexity to detect deepfake videos.
- Initially we used three different CNN networks, smaller in size, for feature extraction. They are (1) Xception, (2) Inception V3, and (3) Resnet50. We compared them and finally selected the Xception network [15] as our CNN module. The detailed work has been discussed in our previous work [40]. In Xception, introduced by Chollet in 2016, depthwise separable convolution has been used which made it accurate and cheaper.
- The feature vector from the CNN is used as the input of a GlobalAveragePooling2D layer with a dropout layer, followed by a fully connected (FC) with a dropout layer and lastly a classification softmax layer. This classifier has been used to detect video.
- Our novelty here is to combine a well known method of computer vision to our simple classifier model for detecting deepfake videos. The existing works have very complex structures [42], [29], [35] for detection. Our main goal is to reduce the computational cost of detection without excessively sacrificing accuracy.
- A novel technique of training without a very large training dataset is reported.
- For training and testing, we primarily used the FF++ and DFDC data sets. We used the compressed deepfake and original videos of the former. These compressed videos have two different compression levels - one is low loss and the other is high loss. The data sets are a good representation of social media scenarios. To obtain a better generalized model, we added another dataset. But to limit the training time we used 1/3 of the DFDC dataset. We compressed the DFDC dataset at three compression levels with the H264 video compressor. Finally we trained our network with this mixed dataset.

### 3.4 The Novelty of the Solution Proposed

Our goal is to obtain a model for detecting social media deepfake videos/images suitable for use at the edge. To achieve that goal we apply a computer vision technique and a simple classifier with the Xception network. The proposed algorithm reduces the computation during detection. By applying the key video frame extraction during data processing we reduced the number of frames significantly and still received a high accuracy. We tried to reduce the data to be processed, as at any edge device memory is a limiting factor.

## 4 Related Prior Works

Identifying manipulated and falsified contents is technically demanding and challenging. In the past two decades in media forensics, substantial work to detect image and video forgery has been done. Most of the solutions proposed for video forensics are for easy manipulations, such as copy-move [17], dropped or duplicated frames [24], or varying interpolation [18]. But use of auto-encoders or GANs has made image/video forgery sophisticated. These computer generated forged videos are hard to detect with previously existing detection techniques. Stacked auto-encoders, CNNs, Long-Short Term Memory (LSTM) networks, or GANS have been explored in detection models to detect video manipulation. Some of the existing works are summarized in Table 1.

Among deep learning solutions, some are temporal feature based and some are based on visual artifacts. In visual-artifact based works, videos are processed frame-by-frame. Each frame contains different features which generate various inconsistencies in the manipulated region of an image. These features are first extracted and then used as input to a deep learning classifier as CNN models can detect these artifacts. The classifiers are ResNet152 [30], VGG16 [48], Inception V3 [51], DenseNet etc. Certain works are associated with detection techniques based on eye blinking rate [37], noting the difference between head pose [55] of an original video and a fake video, and detecting the artifacts of eyes, teeth and face [39]. The human blinking pattern has also been used in another recent paper [31]. A general capsule network based method has been proposed to detect manipulated images and videos [42]. A VGG19 [48] network has been used for latent feature

**Table 1** A Comparative Perspective with Existing Works on Deepfake Video Detection.

| **Works** | **DataSet** | **Model Features** | **Remarks** |
|---|---|---|---|
| Sabir et al. [46] | FaceForensics++ | Use spatio-temporal features of video streams.<br>Bidirectional Recurrent Neural Network (RNN) + DenseNet/ResNet50. | Not applicable to long video clips.<br>Not trained on a large dataset. |
| Güera and Delp [27] | Hollywood-2 Human Actions (HOHA) | Temporal inconsistencies of deepfake video<br>is taken into account.<br>Inception-V3 + LSTM. | Didn't take into account of compressed<br>videos. |
| Li et al. [37] | Closed Eyes in the Wild (CEW) | Used Long Term Recurrent Convolutional Networks.<br>Measured the eye blinking rate.<br>VGG16 + LSTM + FC | Applied to uncompressed videos. |
| Afchur et al. [6] | Downloaded from internet<br>and processed. | Mesonet structures - Meso-4 and MesoInception-4<br>used.<br>2 inception modules + 2 classic convolution layers<br>+ 2 FC layers. | Accuracy is lower for highly<br>compressed video. |
| Li et al. [38] | UADFV and DeepfakeTIMIT | Face warping artifacts. Used 4 CNN models.<br>Measured resolution inconsistency between<br>the warped face area and face. | Compression has not been considered. |
| Matern et al. [39] | A combination<br>of various sources. | Facial texture difference, and<br>missing details in eye and teeth.<br>Logistic regression model and neural network. | Not for compressed video. |
| Nguyen et al. [42] | Four major datasets. | VGG-19 + Capsule Network. | Accuracy is low for highly compressed data. |
| Hashmi et al. [29] | DFDC whole dataset | CNN+LSTM<br>Used facial landmarks and convolutional features | Computation complexity is high.<br>Minimum video length is 10 seconds. Works well for long videos. |
| Kumar et al. [35] | FaceForensics++<br>Celeb-DF | Triplet Architecture.<br>Metric learning approach. | For highly compressed video. |
| Previous work<br>by the authors [40] | FaceForensics++ | Face Artifacts Analysis<br>XceptionNet + Classifier Network | For compressed video.<br>High Accuracy. |
| **Proposed Model and Algorithm** | DFDC and FaceForensics++ | Face Artifacts Analysis with Key Video Frame approach.<br><br>XceptionNet + Classifier Network | For any level of compressed video.<br>Applicable to any video even with only one key video frame.<br>Faster with less computation than authors' previous work. |

extraction along with a capsule network to detect different spoofs, replay attacks etc. Two inception modules along with two classic convolution layers followed by maxpooling layers have been explored in [6]. This approach is at a mesoscopic level. Audio and video parts of a video clip have been used in getting emotion embedding to detect fake videos too [41]. A comparative study among different approaches has been provided in [34] where the authors evaluated existing techniques.

Other than the visual artifact based works, there is another parallel type of work that is prevalent. These are based on temporal features of a video. A combined network of a CNN and LSTM architectures has been explored [27]. The CNN module extracts features from the input image sequence. The feature vector is fed into the LSTM network. Here, the long-short term memory generates a sequence detector from the feature vector. Finally, a fully connected layer classifies the video as manipulated or real. Another combined network was used in a different paper to classify forged videos [46]. A DenseNet structure combined with recurrent neural network (RNN) has been used. A blockchain based ap-

proach to detect forged videos has been proposed by Hasan and Salah [28]. Each video is linked to a smart contract and it has a hierarchical relation to its child video. A video is called pristine if the original smart contract is traced. Unique hashes help to store the data to the InterPlanetary File System (IPFS) peer-to-peer network. This model claims to be extendable to audio or images. In [47], the ownership of a video has been stated by detecting fake video and distinguishing it from real video. Spatio-temporal features has also been used in detecting deepfake videos [21].

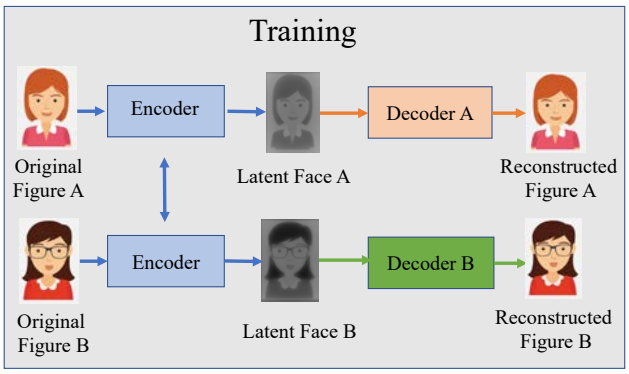
In another recent work [29], deepfake videos are detected using a Convolution-LSTM network. Visual counterfeits have been analyzed. A triplet architecture has been used in detecting deepfake videos at high compression levels [35]. Sharp multiple instance learning has lately been used in detecting partial face attacks in deepfake videos [36]. The performance of the detectors has been improved by clustering face and non-face images and removing the latter [13].

From Table 1, it is clear that there is not much work done for compressed videos which are predominantly used in social media. The FF++ dataset [45] has a huge collection of deepfake videos at different compression levels. The success of XceptionNet on the FF++ dataset motivated us to find a better deep neural model with higher accuracy and not limited to a particular dataset. We assume a video is called manipulated when at least one frame of the video is forged. This simple assumption along with our technique make our video classifier less computation intensive.

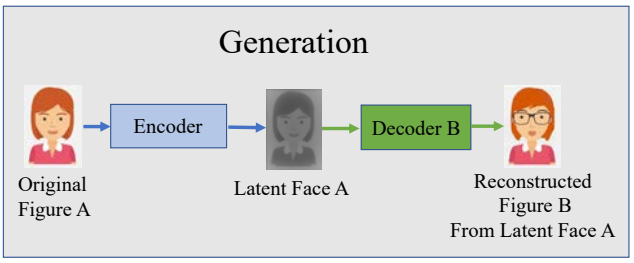
## 5 Why are Deepfakes Hard to Detect?

There are two main ways to create deepfake videos - by autoencoders and by GANs. Both are deep learning methods. Inconsistencies in these videos are not recognizable by bare eyes and thus are hard to detect. Before going into the detecting algorithm, we will discuss these two methods.

By autoencoders: the creation of a deep fake video consists of three steps - extraction, training, and creation. In the extraction process all frames are extracted from a video clip and faces are identified and aligned. An auto-encoder is a combination of an encoder and a decoder. When an image is given to the encoder as input, a latent face, or base vector of lower dimension is created. This vector is then fed into the decoder part of the auto-encoder and the input image is then reconstructed. The shape of the network, like the number of layers and nodes decides the quality of the picture. The description of the network is saved as weights. The training stage is shown in Fig. 3(a). During training those weights are optimized. To make a deepfake video, two sets of autoencoders are needed. One for the original face and the second one for the target video. During training, both encoders share weights for the common features of the source and target faces. There are two separate decoders for the two sets of images. As common features for both image sets are created by a shared encoder, the encoder learns common features by itself and this explains the name. After training is complete, a latent face from image A is passed to decoder B. As a result, decoder B tries to recreate image B from the relative information of image A. Creation of deepfake video frames is shown in Fig. 3(b). This whole process is repeated for all frames or images to make a deepfake video.




(a) Training Phase




(b) Generation Phase

**Fig. 3** Deepfake Video Creation by Autoencoder

By Generative Adversarial Networks (GANs): Creation of deepfake videos using GANs is popular but a GAN is difficult to train. A GAN consists of two neural networks. They are the generator (G) and the discriminator (D). G acts as the encoder of the autoencoder. But as there is a minmax game between G and D, G wants to surpass D by always trying to make a better picture. After good training, the generator generates images of such a quality that the discriminator canâĂŹt distinguish between real and fake images.

There are certain inconsistencies added to the forged video when the fake video is created using autoencoders.

This is because both videos are shot in various environment with different devices. During training the encoder learns the standard deviation (spread) and center (mean) of the latent space distribution separately. Once the latent space vector is created, the decoder comes into play. It tries to recreate the same image as the source by distorting one of the centroids with the standard deviation and adds a random error or noise. The decoder finally generates the image but not exactly as the source. During creation of the fake image, sometimes the tone of skin color does not match well or the edge of spectacles does not fit at the exact position of the nose or ear. We took advantage of these discrepancies in our model as accurately as possible by extensive data processing. Therefore, when the feature extractor extracts features, it gives an accurate feature vector.

## 6 The Proposed Novel Method for Deepfake Detection

Our main goal in this work is to obtain a model which can detect deepfake videos in social media and show the promise to be extended eventually to a model for using at the edge. In this paper, we initially followed our previous paper [40] to choose the Xception network paired with our classifier among ResNet50, InceptionV3 and Xception modules. The framework of our proposed method is shown in Fig. 4.

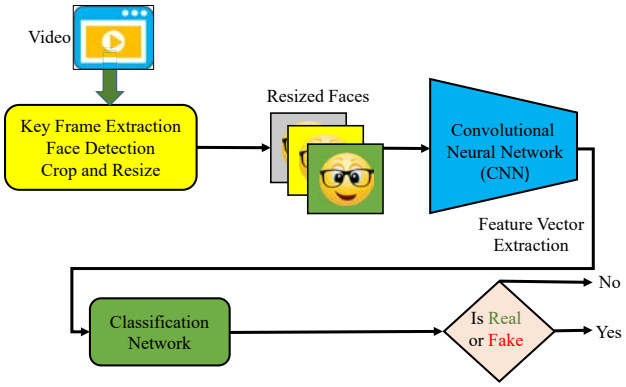



**Fig. 4** A Detailed Representation of the Proposed Model.

The final framework consists of a Xception network and a GlobalAveragePooling2D layer with a dropout layer followed by a fully connected layer with 1024 nodes with dropout and followed by a softmax layer for classification. The CNN module extracts the spatial features and those feature vectors are fed into the classifier part. Finally, a classification result comes out from the classifier. To create a model which does not overfit easily, averagepooling and dropout layers are added to the network accordingly.

The FF++ dataset consists of two different compressed level videos so it is suitable for our experiments. For DFDC, we changed the compression of videos in 3 different levels. Since in social media compressed data is used, frame extraction is done in the compressed video only. No decompression has been done.

*Key Video Frame Extraction:* In a video, there are many elements that don't change in consecutive frames. So, processing each frame and checking for authenticity waste a lot of resources. A key frame or intra-frame or i-frame represents a frame that indicates the beginning or ending of any transition. Subsequent frames contain only the difference in information. To make the model computationally less complex, we extracted only key video frames from videos. As our work mainly focuses on visual artifacts that change with forgery, we assume that only dealing with key frames will be good enough for our model to detect a deepfake video. Fig. 5 shows the key frames from a 20 second video.


**Fig. 5** The Generated Key Video Frames from a 20 second Video.

*Data processing:* Data processing plays a significant role in our work. For the first part of our work we followed the same techniques as before ( [40]). For our newly proposed work, after extracting the key video frames from each video we perform additional data processing. To increase the accuracy of the model we detect all faces and crop the faces from each frame. Finally all frames are normalized and resized as per the input requirement of the CNN module. *Imagesize* is kept at $(299, 299, 3)$ for InceptionV3 and Xception net and $(224, 224, 3)$ for ResNet50. The data processing diagram is shown in Fig. 6.

*Xception Network:* Xception net has been used as feature extractor in our model. It is the extension of Inception architecture but replacing spatial convolution with depthwise separable convolution . The difference between Inception V3 and Xception is the order of $3 \times 3$ spatial channel wise convolution and cross-channel correlation mapping point wise $1 \times 1$ convolutions. The original Xception network has 36 convolution layers which are structured in 14 blocks. Each block, except the first and last, has a linear residual connection.

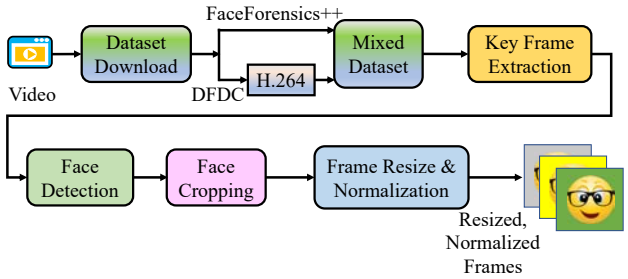



**Fig. 6** The Proposed Flow of Video Processing

It extracts features from all frames and gives a 2048-dimensional feature vector for each frame. It goes to the classifier network. Fig. 7 shows the original Xception network introduced by Chollet.

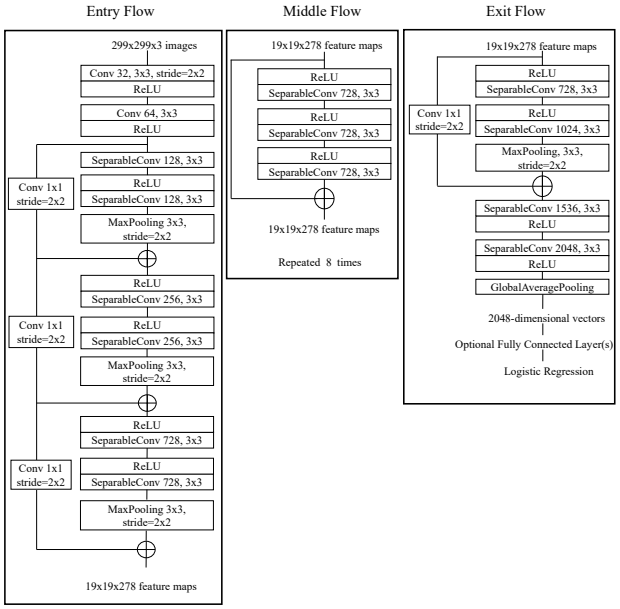



**Fig. 7** Xception Architecture Used as CNN Module in the Proposed Work

*Classification Network:* Fig. 8 shows the classifier network. As the classification network, we chose a combination of layers to get better accuracy. The layers are a GlobalAveragePooling2D layer followed by a dropout layer of 0.5, a fully connected layer with 0.5 dropout and "relu" activation and finally a "softmax" layer which essentially classifies the detected video as real or manipulated. Fig.9 shows how the classifier works.

## 7 The Proposed Method

In the previous Section, the proposed novel technique to detect deepfake videos in social media has been mentioned. The theoretical perspective along with the algorithm have been discussed in the current Section.

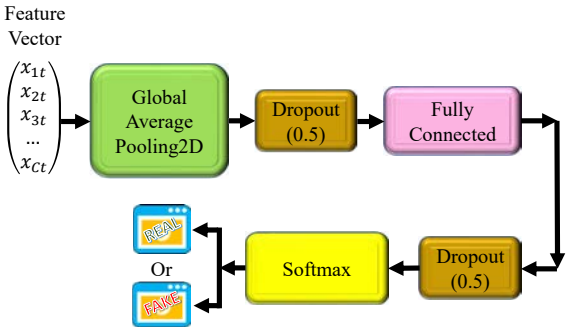



**Fig. 8** Classifier Network Architecture.

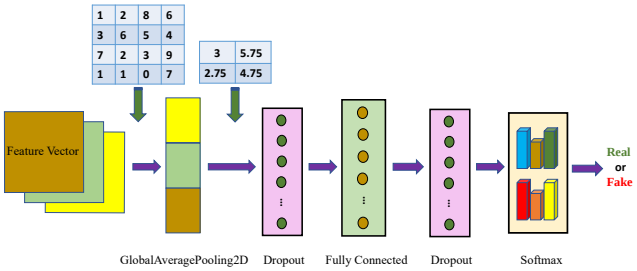



**Fig. 9** Classifier Network Work Flow

### 7.1 A Theoretical Perspective

#### *7.1.1 Depthwise Separable Convolution*

There are three elements in a convolution operation:

- Input image
- Feature detector or Kernel or Filter
- Feature map

The Kernel, or Filter, or Feature Detector is a small matrix of numbers. When it is passed over the input image, new feature maps are generated from the convolution operation between the filter value and the pixel value of the input image at each point $(x, y)$ as the following expression:

$$(I * h)(x, y) = \int_0^x \int_0^y I(x - i, y - j)h(i, j)didj, \qquad (1)$$

where $I$ is the input image and $h$ is the kernel.

The complexity of the convolution operation is expressed as $N \times D_G^2 \times D_K^2 \times M$ where $D_F \times D_F \times M$ is the size of the input image and the filter size is $D_K \times D_K \times M$. $M$ is the number of channels in the input image. The size of the feature matrix is $D_G \times D_G \times M$.

The complexity is decreased in *Depthwise Separable Convolution*. It divides the convolution operation in two parts: (1) Depthwise Convolution - Filtering stage,

and (2) Pointwise Convolution - Combination stage. In depthwise convolution the complexity is $M \times D_G^2 \times D_K^2$ while for pointwise convolution it is $N \times D_G^2 \times D_K^2 \times M$. The overall complexity is can be estimated as the following:

$$\textit{Total Complexity} = M \times D_G^2 \times D_K^2 + N \times D_G^2 \times D_K^2 \times M \tag{2}$$

The relative complexities of the two convolutions is the following:

$$\frac{\textit{Complexity Depthwise Separable Conv.}}{\textit{Complexity Standard Conv.}} = \frac{1}{N} + \frac{1}{D_K^2} \tag{3}$$

It is evident from Eq. (3) that the complexity of standard convolution is much higher than the depthwise separable convolution. It means that the Xception Network provides much faster and cheaper convolution than standard convolution.

#### *7.1.2 GlobalAveragePooling Layer*

It helps to reduce the number of parameters and eventually to minimize overfitting. It down-samples by computing the mean or average of the width and height dimensions of the input. For the Global Average Pooling layer there are no parameters to learn. It takes the average of each feature map for each category of a classification problem and returns a vector which is directly fed into the next layer. It is more robust as it sums up the spatial information.

#### *7.1.3 Dropout Layer*

It is very common for a deep network to overfit. The dropout layer stops the overfitting of a neural network. The least square loss for a single layer linear network with activation function $f(x) = x$ is expressed as the following [10, 49]:

$$E_N = \frac{1}{2}\left(t - \sum_{i=1}^{n} w_i' I_i\right)^2. \tag{4}$$

The least square error of that network with a dropout layer is expressed as the following [10, 49]:

$$E_D = \frac{1}{2}\left(t - \sum_{i=1}^{n} \delta_i w_i I_i,\right)^2. \tag{5}$$

where $\delta_i \sim Bernoulli(p)$. The expectation of the gradient of the dropout network is expressed as:

$$\begin{aligned} E\left[\frac{\partial E_D}{\partial w_i}\right] &= -tp_i I_i + w_i p_i^2 I_i^2 + w_i Var(\delta_i) I_i^2 \\ &\quad + \sum_{j=1, j\neq 1}^{n} w_j p_i p_j I_i I_j, \qquad (6) \\ &= \frac{\partial E_N}{\partial w_i} + w_i p_i (1 - p_i) I_i^2. \qquad (7) \end{aligned}$$

In the above expressions, $w' = p * w$. So if $w' = p * w$, the expectation of the gradient with dropout becomes equal to the gradient of a regularized linear network:

$$E_R = \frac{1}{2}\left(t - \sum_{i=1}^{n} p_i w_i I_i\right)^2 + \sum_{i=1}^{n} p_i (1 - p_i) w_i^2 I_i^2. \tag{8}$$

The above Eq. (8) results in a maximum for $p = 0.5$.

#### *7.1.4 Soft-Max Layer*

In order to predict the class of the video - pristine or manipulated, a softmax layer is used at the end of the network. It takes an $M$-dimensional vector and creates another vector of the same size but with values ranging from 0 to 1 making the sum of the values to 1. If the probability distribution of two classes provided by the softmax layer is $P(y_k)$ for input $y_k$ to the softmax layer, the output $\hat{y}$ from the softmax layer can be predicted by the following expression:

$$\hat{y} = argmax_{k=2} P\left(y_k\right). \tag{9}$$

#### *7.1.5 Training Loss*

During training, we minimize the *Categorical Cross Entropy Loss* to get optimal parameters of the network to best predict the class. It is the measure of performance of a classification model whose output is a probability, and ranges from 0 to 1. In binary classification like our case the cross entropy loss is expressed as:

$$\mathcal{L} = -\left(y \log(\hat{y}) + (1 - y) \log(1 - \hat{y})\right). \tag{10}$$

We use the *Adam* optimizer to minimize the loss stated in Eq. (10). One mini batch is processed at each iteration to get optimal parameters. After several epochs, when the loss function $\mathcal{L}$ is optimized, network parameters are learned to their optimal value.

**Algorithm 1:** Steps to Detect DeepFake Video.

**Input:**Test video $v$, Model $\tilde{M}$
**Output:**Label $tag$
Declare and initialize $frames$, $f$, $face$, and $resface$ to 0
Assign total number of frames, a particular frame , cropped face respect to the frame $f$, and resized face respect to the face $face$ to the initialized variables respectively
Declare and initialize $realtag$ and $faketag$ to 0
Assign real probability and fake probability after prediction to these variables respectively
Set $tag = False$
Extract **all frames** from the test video $v$
Save the extracted frames in $frames$
**for** $f \in frames$ **do**
    Detect the face for $f$
    Crop the face and Save it in $face$
    Resize the image and Save it in $resface$
    Load the Model $\tilde{M}$
    Predict $resface$
    Set $realtag$ to real probability of the prediction
    Set $faketag$ to fake probability of the prediction
    **if** $realtag \gg faketag$ **then**
        **continue**
    **else**
        Set $tag = True$
        Consider the video as Fake
        **break**

**Algorithm 2:** Steps to Detect DeepFake Video using Key Video Frames Approach.

**Input:**Test video $v$, Model $\tilde{M}$
**Output:**Label $tag$
Declare and initialize $frames$, $f$, $face$, and $resface$ to 0
Assign total number of Key Video Frames, a particular key video frame , cropped face respect to the key video frame $f$, and resized face respect to the face $face$ to the initialized variables respectively
Declare and initialize $realtag$ and $faketag$ to 0
Assign real probability and fake probability after prediction to these variables respectively
Set $tag = False$
Extract **key video frames** from the video $v$.
Save the extracted frames in $frames$.
**for** $f \in frames$ **do**
    Detect the face for $f$
    Crop the face and Save it in $face$
    Resize the image and Save it in $resface$
    Load the Model $\tilde{M}$
    Predict $resface$
    Set $realtag$ to real probability of the prediction
    Set $faketag$ to fake probability of the prediction
    **if** $realtag \gg faketag$ **then**
        **continue**
    **else**
        Set $tag = True$
        Consider the video as Fake
        **break**

### 7.2 Details of the Proposed Algorithms

In our initial Algorithm 1 deepfake videos are checked frame by frame [40]. The accuracy obtained is high. But we had to process a very large number of frames. Our proposed algorithm originates from this necessity. In the current paper, we also propose Algorithm 2 in detecting deepfake videos to reduce the computation.

- The novelty of our first algorithm is to make the complexity of detecting forged video small. The time complexity is $\mathbf{O}(n)$ where $n$ is the number of frames extracted from the video.
- The reason for proposing Algorithm 2 is to further reduce the complexity. As key frame extraction reduces the number of extracted frames from a video largely, the time complexity is reduced too.

## 8 Experimental Validation

In this Section we report the experiments and corresponding results. We start with the details of the dataset, experimental parameters and finally analyze the results.

### 8.1 Datasets

There are several datasets available for image manipulation but not for video, such as the "Dresden image database" [25], the MFCC_F200 dataset [8], the First IEEE Image Forensics Challenge Dataset, and the Wild Web Dataset [56]. Recently, two major datasets have been created for deepfake detection research. Google produced a deepfake detection data set teaming up with Jigsaw. The dataset has been added in the FaceForensics benchmark of the Technical University of Munich and the University Federico II of Naples in September, 2019. In the same year, Facebook Inc. in collaboration with Microsoft AI, Microsoft Corp and academic institutions such as Cornell, MIT, University of Oxford, etc. has started the Deepfake Detection Challenge (DFDC) [20]. The FF++ [44] and DFDC data sets [19] are a good start to establish models for deepfake video detection.

For selecting the CNN module, we trained the network with only FF++ data at compression level $c$=23, as shown in Table 2. But to make a generalized model, during the final part of our work we trained our neural network model with mixed compression level dataset and we had to construct the dataset as in Table 3. To

train our model we construct a mixed dataset
ing of DFDC and FF++ dataset. We focused
compressed dataset as any image/video loses c
in other terms shows losses as compression ir
FF++ has two different compression levels vi
represents a realistic scenario for social media. W
the compression level of DFDC dataset in mal
dataset. The dataset details are shown in Tabl

**Table 2** Dataset Details for Initial Work

| Dataset Division | No. of Original Videos ( $c$=23) | No. of Manipu Videos ($c$=23) |
|---|---|---|
| Train | 800 | 800 |
| Valid | 100 | 100 |
| Test | 100 | 100 |

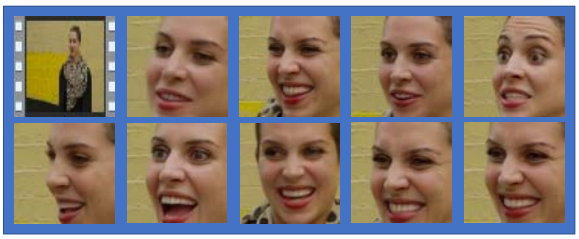
**Table 3** Dataset Details for Final Work

| Dataset | No. of Original Videos($c$=15, $c$=23, $c$=40) | No. of Manipulated Videos($c$=15, $c$=23, $c$=40) |
|---|---|---|
| FF++ | 2000 | 2000 |
| DFDC | 5773 | 5765 |

– *FaceForensics++ (FF++) dataset:* For our work, we used 2000 deepfake videos and 2000 original videos at different compression levels. As the videos uploaded in social media are compressed, FaceForensics++ dataset is a good representative of the social media scenario. We used two compression level video sets - one with quantization parameter or constant rate factor 23 and the other at 40.
– *Deepfake Detection Challenge (DFDC) dataset:* We used part of 470GB dataset - 5765 manipulated videos and 5773 original videos. We changed the compression levels to three levels $c = 15$, $c = 23$, and $c = 40$. We kept the number of videos for low level compression $c = 15$ minimum. As loss increases with the compression level, we wanted to train our network more on $c = 23$ and $c = 40$ than $c = 15$ videos. We changed the compression levels with an H.264 encoder using FFmpeg software [11].

We constructed our data set with 7773 pristine and 7765 forged videos of different compression levels. We kept the number of videos almost the same for each class - manipulated and real, to negate any kind of preference or bias in data. We kept 600 mixed compression original and manipulated videos aside for testing the accuracy of the model and the rest are used for training and validation. Fig. 10(a) shows the key video frames from

10 Sec vid



24 Sec Video

(b) From a 24 sec fake video

**Fig. 10** Key Video Frames from different length videos

## 8.2 Experimental Setup

In this section, we discuss the implementation set up. The whole work consists of two parts. In the first part, we chose our feature extractor and in the second part we introduced a unique way to detect deepfake videos using a key frame extraction technique to reduce the computations. The first part was trained on a smaller dataset.

*Transfer Learning:* We used transfer learning for better accuracy and to save training time. A pre-trained model approach was taken. Initially Resnet50, InceptionV3 and Xception net have been chosen as the feature extractor which are trained on Imagenet dataset. So they already have learned to detect basic and general features of images as they were trained on 1000 classes of 1,000,000 images. Lower level layers extract basic features like lines or edges whereas middle or higher layers extract more complex and abstract features and features defining classification.

To train our network first we trained the classifier keeping the weights of feature extractor frozen and then fine tuned the whole network from end-to-end. We repeated the whole process for our three CNN modules and finally chose the Xception network as our feature extractor. The overall work flow is presented in Fig. 11. Fig. 12 shows the steps followed in our work for training and validation. Table 4 shows the number of frames used for training and validation purposes our work.

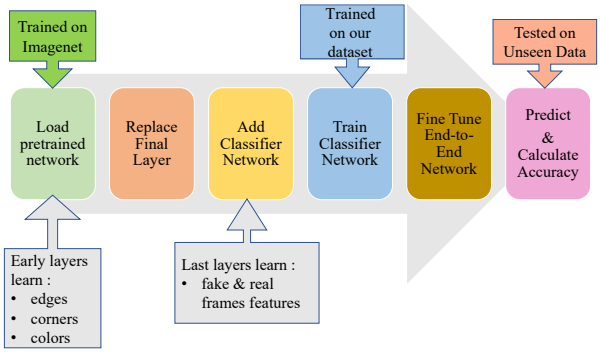



**Fig. 11** End-to-End Work Flow.

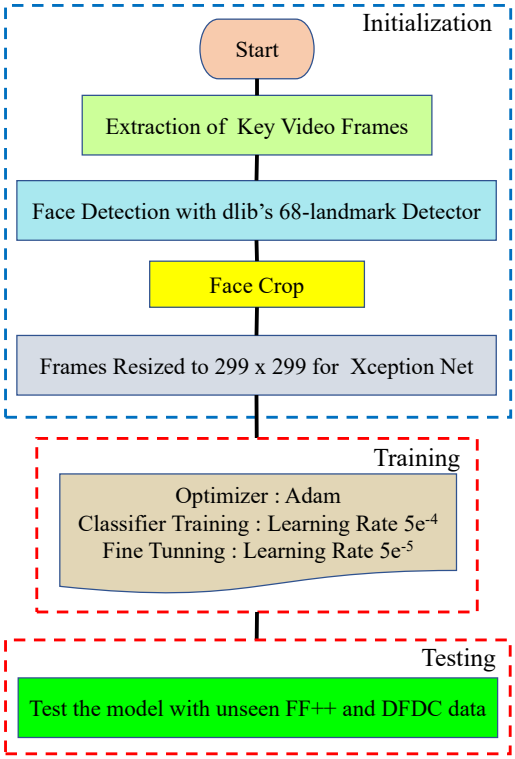



**Fig. 12** The specific steps followed for testing and validation purposes.

**Table 4** Frames Details for Training and Validation

| Division | **No. of Frames** During **Part-1** | **No. of Frames** During **Part-2** |
|---|---|---|
| Train | 124959 | 49373 |
| Valid | 31240 | 12345 |

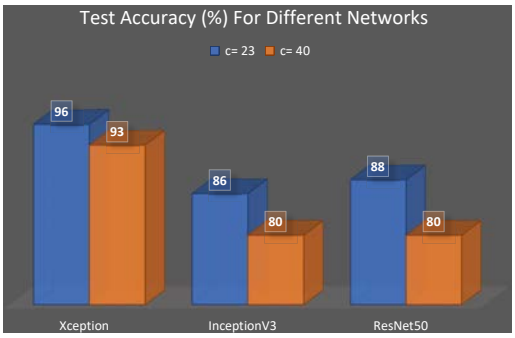
*Implementation Details:* We implemented our proposed framework in Keras with the TensorFlow backend. FFmpeg is used to clip the videos and 68-landmarks in the dlib package for face detection. For training we used a Tesla T4 GPU with 64GB Memory. A GeForce RTX 2060 is used to evaluate the model.

### 8.3 Experimental Verification

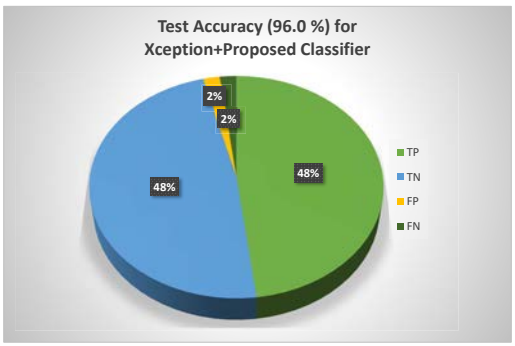
Initially we verified our model with unseen data from FF++. Our model with Xception net gave the best accuracy among all three CNN modules. The accuracy for compression level $c$=23 was better than that of $c$=40. Once we finalized our feature extractor, we trained the model applying our newly proposed Algorithm 2 along with our customized dataset from FF++ and DFDC and verified it with unseen data from FF++ and DFDC test dataset. We changed the compression levels of the DFDC test dataset to represent social media videos.

### 8.4 Results

Fig. 13(a) shows the accuracy vs different CNN networks. It is clear that the Xception net performed better with our classifier. Once the feature extractor was chosen, we moved to the final work. Fig. 14(b), 14(c), and 14(d) show the output of different layers of Xception net for the key video frame (Fig. 14(a)).




(a) Test Accuracy for Different Networks

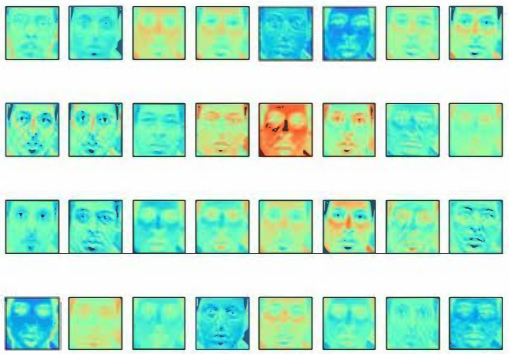



(b) Test Accuracy Calculation

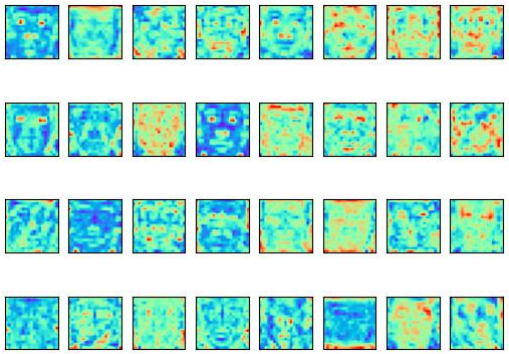
**Fig. 13** Results of the First Experiment with only FaceForensics ++ dataset.

In our initial experiment we tested 200 videos of FF++ unseen data and chose our feature extractor. We obtained 96% accuracy for the videos with compression level $c$ = 23. Then we tested the model with

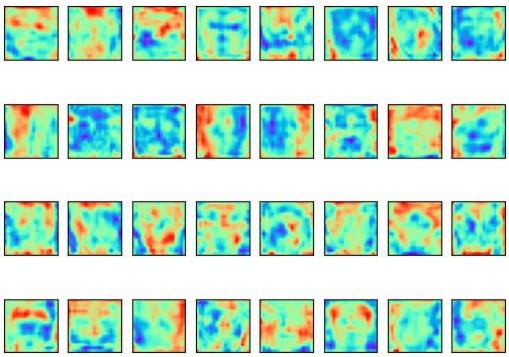
(a) Fake Test Image Frame



(b) What does layer 10 of Xception see?



(c) What does layer 45 of Xception see?



(d) What does layer 90 of Xception see?

**Fig. 14** Sample View of CNN layer output.

our newly proposed Algorithm 2 and two sets of data - first with the same 200 videos from FF++ we used for testing initially and achieved 98.5% accuracy. Then we tested the model with 600 mixed compression videos from FF++ and DFDC test dataset. Most of them are highly compressed ($c$=40). We were able to achieve accuracy of 92.33% even with high loss videos. The results are shown in Fig. 16.

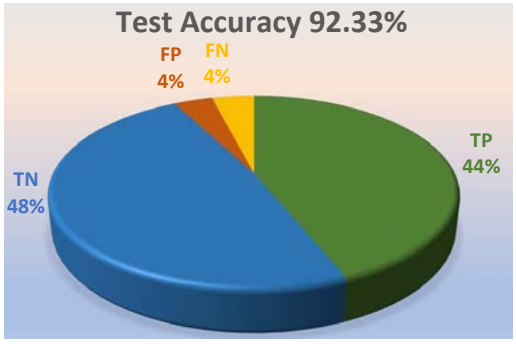
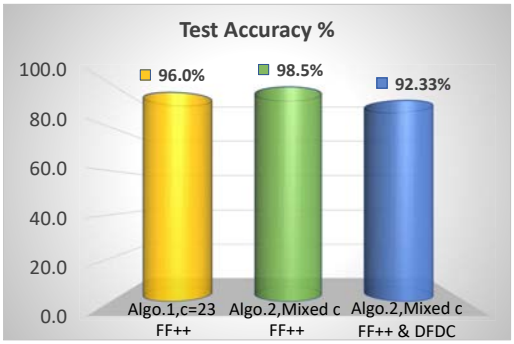
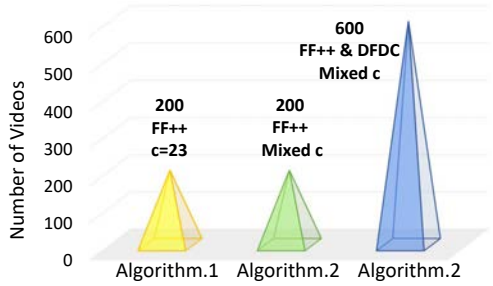



**Fig. 15** Test Accuracy Calculation for Final Experiment.




(a) Test Accuracy Plot




(b) Number of Videos for the Experiments

**Fig. 16** Comparison of Results between two Experiments.

## 8.5 Analysis of Results

As our case is a binary classification, to visualize the performance of our model we define our confusion matrix as in Table 5.

The first performance metric we can derive from the confusion matrix is accuracy as defined in Eq. (11):

$$Accuracy = \left( \frac{TP + TN}{TP + TN + FP + FN} \right). \quad (11)$$

**Table 5** Confusion Matrix - Definition of TP, TN, FP and FN.

| | |
|---|---|
| True Positive (**TP**):<br>Reality : **Fake(1)**<br>Model predicted : **Fake(1)** | False Negative (**FN**):<br>Reality : **Fake (1)**<br>Model predicted : **Real(0)** |
| False Positive (**FP**):<br>Reality : **Real(0)**<br>Model predicted : **Fake(1)** | True Negative (**TN**):<br>Reality : **Real(0)**<br>Model predicted : **Real(0)** |

To calculate the accuracy of the model we follow Table. 5. The detailed results for calculating test accuracy are shown in Table 6.

**Table 6** Data to Calculate Test Accuracy

| Algorithm + Compression+ Test Videos | **CNN** + Proposed Classifier | Number of Test Data TP | TN | FP | FN |
|---|---|---|---|---|---|
| Algorithm.1 | ResNet50 | 80 | 96 | 04 | 20 |
| + c=23 | InceptionV3 | 84 | 88 | 12 | 16 |
| + 200 (FF++) videos | **Xception** | 96 | 96 | 04 | 04 |
| **Algorithm.2 + Mixed c + 200 (FF++) videos** | **Xception** | 98 | 99 | 01 | 02 |
| **Algorithm.2 + Mixed c + 600 (FF++ & DFDC) videos** | **Xception** | 276 | 278 | 22 | 24 |

For the initial part, all three CNN modules have been paired with our classifier network. Xception net combined with our classifier gave the best accuracy of 96.00% for compression level $c = 23$. Test accuracy for both experiments is reported in Table 7. The number of FN is lowest in the case of Xception. It is 1%, leading test accuracy close to train and validation accuracy.

**Table 7** Test Accuracy

| Testing Data | No. of Videos | Algorithm | Compression Levels | Accuracy |
|---|---|---|---|---|
| FF++ | 200 | Algorithm.1 | $c = 23$ | 96.00 |
| FF++ | 200 | Algorithm.2 | $c \leq 40$ | 98.50 |
| FF++ DFDC | 600 | Algorithm.2 | $c \leq 40$ | 92.33 |

Test accuracy for our model applying Algorithm 2 is shown in Fig. 15. *Precision*, *Recall* and *F1-score* for our model are defined by the following expressions:

$$Precision = \left(\frac{TP}{TP + FP}\right) \tag{12}$$

$$Recall = \left(\frac{TP}{TP + FN}\right) \tag{13}$$

$$F1 - score = \left(\frac{2}{\frac{1}{Precision} + \frac{1}{Recall}}\right) \tag{14}$$

For our classification model, the above metrics are calculated using Table 6 and are shown in Fig. 17 and Table 8.

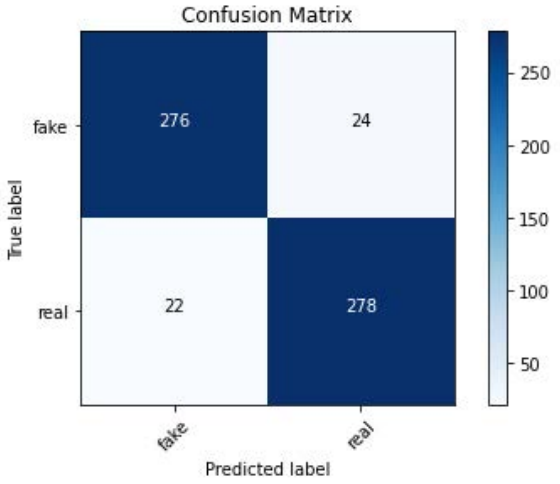



**Fig. 17** Metric Calculation

Our Algorithm 1 is applicable to any length video as we process frames at 24 fps. In most fake videos, since only the face is changed the number of key frames is low. Our Algorithm 2 can detect fake videos even if only one key frame is extracted from the testing video but its accuracy increases enormously (almost all results were correct) if the video has more than one key frame. Our reported accuracy for Algorithm 2 considers all cases even if the video contains only one key frame. That is why we report accuracy as 92.33%.

If the video is very hazy, our model might not produce accurate results. The number of FN is mostly contributed by the hazy fake videos. The number of hazy videos in the training and validation dataset was not sufficient for our model to learn and resulted in uncertain prediction for those videos.

The accuracy of our model was lower when we added DFDC dataset with FF++ dataset because we train our model with only partial DFDC dataset. We believe that, as DFDC dataset is a vast dataset, training our model with the entire dataset would have resulted in better accuracy.

**Table 8** Accuracy, Precision, Recall and F1-score Calculation for Test Videos

| Test Videos | Algorithm | Model | Accuracy% | Precision | Recall | F1-score |
|---|---|---|---|---|---|---|
| 200 FF++ Videos | Algorithm.1 | ResNet50 + Classifier | 88.00 | 0.95 | 0.80 | 0.87 |
| | | InceptionV3 + Classifier | 86.00 | 0.89 | 0.88 | 0.88 |
| | | Xception + Classifier | **96.00** | 0.96 | 0.96 | 0.96 |
| 200 FF++ Videos | Algorithm.2 | **Xception + Classifier** | **98.50** | 0.99 | 0.98 | 0.98 |
| 600 FF++ & DFDC Videos | Algorithm.2 | **Xception + Classifier** | **92.33** | 0.93 | 0.92 | 0.92 |

## 8.6 Comparisons

We compared our results with existing results [29, 35, 45]. The detailed comparison is shown in Table 9. Fig. 18 shows a comparative view between our and other existing works. We achieved 98.5% and 92.33% accuracy with our proposed algorithm for two different test dataset videos. We believe incorporating more manipulated videos with different performers, and different light and noise condition in the training dataset will increase the performance of our model.

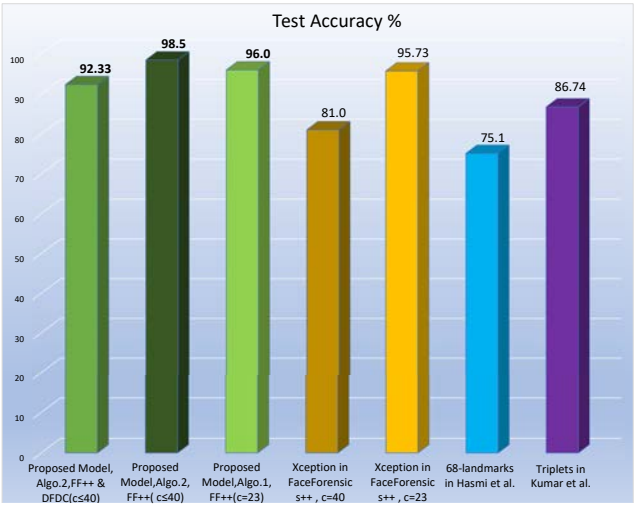



**Fig. 18** Accuracy Comparison.

**Table 9** Performance Comparison of Xception Network paired with Proposed Classifier

| Network | Test Dataset (Unseen) | Compression Level | Accuracy (%) |
|---|---|---|---|
| **Xception + Proposed Classifier + Algo.2 + 68 landmarks** | FF++ DFDC | Any (checked upto c=40) | **92.33** |
| **Xception + Proposed Classifier + Algo.2 + 68 landmarks** | FF++ | Any (checked upto c=40) | **98.50** |
| **Xception + Proposed Classifier + Algo.1 + 68 landmarks** [40] | FF++ | c=23 | **96.00** |
| Xception in FF++ Paper [45] | FF++ | c=40 | 81.00 |
| | | c=23 | 95.73 |
| CNN + LSTM + 68-landmarks [29] | DFDC | NA | 75.10 |
| Triplets(Semi-hard) [35] | FF++ | c=40 | 86.74 |

## 9 Conclusions and Future Works

In this paper we present a deep learning based approach to detect deepfake videos in social media with low computational burden. Our final goal is to achieve an edge based fake video detecting method. We believe that the proposed algorithm is the first founding step for achieving that.

Initially we chose three CNN modules as feature extractor and finally selected the Xception network as the feature extractor of our model. First we classified pristine and manipulated videos proposing an algorithm which processes every frame of a video. We worked with a specific compression level $c$=23 as it is in the mid range low loss compression level. We achieved a high accuracy based on our algorithm. The complexity of the algorithm is proportional to the number of frames extracted from the video. To make the number of computations smaller, we proposed a second algorithm where we utilized the key video frame technique of computer vision to reduce the number of computations. We evaluated with a much bigger unseen dataset and were able to achieve good accuracy for highly compressed high loss data.

In this paper:

- We proposed a deep neural method for detecting social media deepfake videos.
- An algorithm which cuts down the computational burden significantly has been proposed.

– We avoided training with enormous amounts of data even though we accommodated a large number of videos.
– We achieved high accuracy even for highly compressed video.

We evaluated our algorithm with the well established FF++ and DFDC datasets. As our algorithm reduces the computations significantly, we believe that it can be deployed at edge device with appropriate modifications.

Embedded deep learning is a growing field. Serious demand for various application domains is pushing today's cloud dependent deep learning area. As our algorithm detects fake videos by detecting key video frames, it substantially reduces the computation. So, it is one step forward to deploy a video detecting model at an edge device. But due to the memory limitation deep neural network structure is large to fit at the edge devices. So, reducing the run-time memory and the model size would be a great effort as the future work. Our immediate future research is focused on deepfake detection in other media like National IDs.

**Acknowledgements** This article is an extended version of our previous conference paper presented at [40].

## Compliance with Ethical Standards

The authors declare that they have no conflict of interest and there was no human or animal testing or participation involved in this research. All data were obtained from public domain sources.

## References

1. DeepFaceLab. Web. URL https://github.com/iperov/DeepFaceLab. Last Accessed on 19 January 2021
2. Deepfake Video. Web. URL https://edition.cnn.com/2019/06/11/tech/zuckerberg-deepfake/index.html. Last Access on 19 January 2021
3. DFaker. Web. URL https://github.com/dfaker/df. Last Accessed on 19 January 2021
4. Faceswap. Web. URL https://github.com/deepfakes/faceswap. Accessed 19 January 2021
5. Faceswap. Web. URL https://faceswap.dev/. Last Accessed on 19 January 2021
6. Afchar, D., Nozick, V., Yamagishi, J., Echizen, I.: MesoNet: a compact facial video forgery detection network. In: Proc. IEEE International Workshop on Information Forensics and Security (WIFS), pp. 1–7 (2018)
7. Allcott, H., Gentzkow, M.: Social media and fake news in the 2016 election. Journal of Economic Perspectives **31**(2), 211–36 (Spring 2017)
8. Amerini, I., Ballan, L., Caldelli, R., Del Bimbo, A., Serra, G.: A sift-based forensic method for copy-move attack detection and transformation recovery. IEEE Transactions on Information Forensics and Security **6**(3), 1099–1110 (2011)
9. Baccarella, C., Wagner, T., Kietzmann, J., McCarthy, I.: Social media? it's serious! understanding the dark side of social media. European Management Journal **36**, 431–438 (2018)
10. Baldi, P., Sadowski, P.J.: Understanding dropout. In: C.J.C. Burges, L. Bottou, M. Welling, Z. Ghahramani, K.Q. Weinberger (eds.) Advances in Neural Information Processing Systems, vol. 26, pp. 2814–2822 (2013)
11. Bellard, F., Niedermayer, M.: Ffmpeg. Web (2012). URL http://ffmpeg.org. Last Accessed on 19 January 2021
12. Bregler, C., Covell, M., Slaney, M.: Video rewrite: Driving visual speech with audio. In: Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques, pp. 353–360 (1997)
13. Charitidis, P., Kordopatis-Zilos, G., Papadopoulos, S., Kompatsiaris, I.: Investigating the impact of pre-processing and prediction aggregation on the deepfake detection task. arXiv (2006.07084) (2020)
14. Chesney, R., Citron, D.: Deepfakes: A looming crisis for national security, democracy and privacy? Web (2018). URL https://www.lawfareblog.com/deepfakes-looming-crisis-national-security-democracy-and-privacy
15. Chollet, F.: Xception: Deep learning with depthwise separable convolutions. CoRR **abs/1610.02357** (2016). URL http://arxiv.org/abs/1610.02357
16. Chou, H.T.G., Edge, N.: They Are Happier and Having Better Lives than I Am: The Impact of Using Facebook on Perceptions of Others' Lives. Cyberpsychology, Behavior, and Social Networking **15**(2), 117–121 (2012)
17. D'Amiano, L., Cozzolino, D., Poggi, G., Verdoliva, L.: A PatchMatch-based dense-field algorithm for video copy-move detection and localization. IEEE Transactions on Circuits and Systems for Video Technology **29**(3), 669–682 (2019)
18. Ding, X., Yang, G., Li, R., Zhang, L., Li, Y., Sun, X.: Identification of motion-compensated frame rate up-conversion based on residual signals. IEEE Transactions on Circuits and Systems for Video Technology **28**(7), 1497–1512 (2018)
19. Dolhansky, B., Bitton, J., Pflaum, B., Lu, J., Howes, R., Wang, M., Ferrer, C.C.: The deepfake detection challenge dataset. arXiv **2006.07397** (2020)
20. Dolhansky, B., Howes, R., Pflaum, B., Baram, N., Canton-Ferrer, C.: The deepfake detection challenge (dfdc) preview dataset. ArXiv **abs/1910.08854** (2019)
21. Ganiyusufoglu, I., Ngo, L.M., Savov, N., Karaoglu, S., Gevers, T.: Spatio-temporal features for generalized detection of deepfake videos (2020)
22. Garrido, P., Valgaerts, L., Rehmsen, O., Thormaehlen, T., Perez, P., Theobalt, C.: Automatic face reenactment. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition, pp. 4217–4224 (2014)
23. Gerstner, E.: Face/off: "DeepFake" face swaps and privacy laws. Defense Counsel Journal **87**(1) (2020)
24. Gironi, A., Fontani, M., Bianchi, T., Piva, A., Barni, M.: A video forensic technique for detecting frame deletion and insertion. In: Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 6226–6230 (2014)
25. Gloe, T., Böhme, R.: The "Dresden image database" for benchmarking digital image forensics. In: Proceedings of the ACM Symposium on Applied Computing, pp. 1584–1590 (2010)

26. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. In: Proc. Advances in Neural Information Processing Systems, pp. 2672–2680 (2014)
27. Güera, D., Delp, E.J.: Deepfake video detection using recurrent neural networks. In: Proc. 15th IEEE International Conference on Advanced Video and Signal Based Surveillance, pp. 1–6 (2018)
28. Hasan, H.R., Salah, K.: Combating deepfake videos using blockchain and smart contracts. IEEE Access **7**, 41596–41606 (2019)
29. Hashmi, M.F., Ashish, B.K.K., Keskar, A.G., Bokde, N.D., Yoon, J.H., Geem, Z.W.: An exploratory analysis on visual counterfeits using conv-lstm hybrid architecture. IEEE Access **8**, 101293–101308 (2020)
30. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 770–778 (2016)
31. Jung, T., Kim, S., Kim, K.: DeepVision: Deepfakes Detection Using Human Eye Blinking Pattern. IEEE Access **8**, 83144–83154 (2020). DOI 10.1109/ACCESS.2020.2988660
32. Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., Aila, T.: Analyzing and improving the image quality of stylegan. In: Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 8107–8116 (2020)
33. Kietzmann, J., Lee, L.W., McCarthy, I.P., Kietzmann, T.C.: Deepfakes: Trick or treat? Business Horizons **63**(2), pp. 135 – 146 (2020)
34. Korshunov, P., Marcel, S.: Deepfakes: a new threat to face recognition? assessment and detection (2018)
35. Kumar, A., Bhavsar, A., Verma, R.: Detecting deepfakes with metric learning. In: 2020 8th International Workshop on Biometrics and Forensics (IWBF), pp. 1–6 (2020)
36. Li, X., Lang, Y., Chen, Y., Mao, X., He, Y., Wang, S., Xue, H., Lu, Q.: Sharp multiple instance learning for deepfake video detection. In: Proceedings of the 28th ACM International Conference on Multimedia, pp. 1864–1872 (2020)
37. Li, Y., Chang, M., Lyu, S.: In Ictu Oculi: Exposing AI created fake videos by detecting eye blinking. In: Proc. IEEE International Workshop on Information Forensics and Security, pp. 1–7 (2018)
38. Li, Y., Lyu, S.: Exposing deepfake videos by detecting face warping artifacts. CoRR **abs/1811.00656** (2018). URL http://arxiv.org/abs/1811.00656
39. Matern, F., Riess, C., Stamminger, M.: Exploiting visual artifacts to expose deepfakes and face manipulations. In: Proc. IEEE Winter Applications of Computer Vision Workshops, pp. 83–92 (2019)
40. Mitra, A., Mohanty, S.P., Corcoran, P., Kougianos, E.: A Novel Machine Learning based Method for Deepfake Video Detection in Social Media. In: Proceedings of the 6th IEEE International Symposium on Smart Electronic Systems (iSES), p. In Press (2020)
41. Mittal, T., Bhattacharya, U., Chandra, R., Bera, A., Manocha, D.: Emotions don't lie: An audio-visual deepfake detection method using affective cues. In: Proceedings of the 28th ACM International Conference on Multimedia, pp. 2823–2832 (2020)
42. Nguyen, H.H., Yamagishi, J., Echizen, I.: Capsule-forensics: Using capsule networks to detect forged images and videos. In: Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 2307–2311 (2019)
43. Reid, S.: The deepfake dilemma: Reconciling privacy and first amendment protections. University of Pennsylvania Journal of Constitutional Law, Forthcoming (June 26,2020). URL https://ssrn.com/abstract=3636464
44. Rössler, A., Cozzolino, D., Verdoliva, L., Riess, C., Thies, J., Nießner, M.: Faceforensics: A large-scale video dataset for forgery detection in human faces. ArXiv **abs/1803.09179** (2018)
45. Rossler, A., Cozzolino, D., Verdoliva, L., Riess, C., Thies, J., Niessner, M.: FaceForensics++: Learning to detect manipulated facial images. In: Proc. IEEE International Conference on Computer Vision, pp. 1–11 (2019)
46. Sabir, E., Cheng, J., Jaiswal, A., AbdAlmageed, W., Masi, I., Natarajan, P.: Recurrent convolutional strategies for face manipulation detection in videos. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, pp. 80–87 (2019)
47. Sethi, L., Dave, A., Bhagwani, R., Biwalkar, A.: Video security against deepfakes and other forgeries. Journal of Discrete Mathematical Sciences and Cryptography **23**, 349–363 (2020)
48. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv (1409.1556) (2014)
49. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: A simple way to prevent neural networks from overfitting. J. Mach. Learn. Res. **15**(1), 1929–1958 (2014)
50. Suwajanakorn, S., Seitz, S.M., Kemelmacher-Shlizerman, I.: Synthesizing obama: learning lip sync from audio. ACM Transactions on Graphics (TOG) **36**(4) (2017)
51. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. arXiv (1512.00567) (2015)
52. Tewari, A., Zollhofer, M., Kim, H., Garrido, P., Bernard, F., Perez, P., Theobalt, C.: MoFA: Model-based deep convolutional face autoencoder for unsupervised monocular reconstruction. In: Proc. IEEE International Conference on Computer Vision (ICCV), pp. 3735–3744 (2017)
53. Thies, J., Zollhofer, M., Niessner, M., Valgaerts, L., Stamminger, M., Theobalt, C.: Realtime expression transfer for facial reenactment. In: ACM Transactions on Graphics (TOG) - Proceedings of ACM SIGGRAPH Asia 2015, *Art No.183*, vol. 34 (2015)
54. Westerlund, M.: The emergence of deepfake technology: A review. Technology Innovation Management Review **9**, 39–52 (2019). DOI 10.22215/timreview/1282
55. Yang, X., Li, Y., Lyu, S.: Exposing deep fakes using inconsistent head poses. In: Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 8261–8265 (2019)
56. Zampoglou, M., Papadopoulos, S., Kompatsiaris, Y.: Detecting image splicing in the wild (web). In: Proc. IEEE International Conference on Multimedia Expo Workshops (ICMEW), pp. 1–6 (2015)
57. Zhao, Y., Chen, C.: Unpaired image-to-image translation via latent energy transport. arXiv (2012.00649) (2020)
58. Zhu, J., Park, T., Isola, P., Efros, A.A.: Unpaired image-to-image translation using cycle-consistent adversarial networks. In: Proc. IEEE International Conference on Computer Vision (ICCV), pp. 2242–2251 (2017)