Employing optical flow on convolutional recurrent structures for deepfake detection

Akash Chintha
ac1864@rit.edu

Employing optical flow on convolutional recurrent structures for deepfake detection

by

Akash Chintha

June 13, 2020

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of

Master of Science in Computer Engineering

Department of Computer Engineering

Kate Gleason College of Engineering

Rochester Institute of Technology

Approved By:

Dr. Raymond Ptucha	Date
Primary Advisor - RIT Dept. of Computer Engineering	
Dr. Matthew Wright	Date
Secondary Advisor - RIT Dept. of Computing and Information Sciences	
Dr. Shanchieh Yang	Date
Secondary Advisor - RIT Dept. of Computer Engineering	

ACKNOWLEDGEMENTS

I would like to take this opportunity to thank Dr. Raymond Ptucha and Dr. Matthew Wright for continuous guidance and support in my research. I am also thankful to Dr. Shanchieh Yang for showing interest in my work and being a part of the thesis committee. I would like to thank my coresearchers Saniat Sohrawardi, Aishwarya Rao, Bao Thai, Akhil Santha, and Kartavya Bhatt for assisting me with my thesis research. Finally, I would like to thank my parents and my brother for being my towers of strength throughout my career with their unconditional love and support.

Contents

1 Introduction **10**

2 Literature Review **13**

2.1 Deepfake Generation 13

2.2 Deepfake Detection 16

3 Background **20**

3.1 CNNs for feature extraction 20

3.2 Long Short-Term Memory (LSTM) 23

3.3 Optical Flow 25

3.4 Edge Detection 25

4 Proposed Deepfake Detection Architecture **27**

4.1 Preprocessing 27

4.1.1 Face Extraction 27

4.1.2 Optical flow and Edges 28

4.2 Architectures 29

4.2.1 CNN+Temporal Model 29

4.2.2 RGB+EF XcepTemporal Model 30

4.2.3 Loss Functions 31

4.2.4 Variants 34

5 Results **38**

5.1 Datasets 38

5.2 CNN+Temporal Model 40

5.2.1 Baselines 40

5.2.2 Training 41

5.2.3 Within-Domain Results 42

5.2.4 Cross-Domain Results 45

5.2.5 Compression 45

5.2.6 Transfer learning / Domain Adaptation. 51

5.2.7 Ablation Study 52

5.3 RGB+EF XcepTemporal Model 53

5.3.1 Within-Domain Results 53

5.3.2 Cross-Domain Results 55

5.3.3 Immunizability on Adversarial Attacks 56

6 Conclusion **60**

6.1 Future Work 60

List of Figures

1 An overview of the problem space. Audio and visual tracks are extracted and processed through spoof and deepfake detection models. 11

2 XceptionNet as a feature extraction model, converting each video frame to a latent vector embedding. 21

3 EfficientNet baseline archiecture [76]. 22

4 An LSTM unit (adapted from [26]) 23

5 The EdgeFlow (EF) input shown on the right is generated using the test sample on the left. 28

6 The CNN+Temporal model. Faces are first extracted and normalized to a canonical representation. These canonical images are passed into a convolutional recurrent model to make a prediction. μ and σ^2 represent the mean and variance of the distribution in a two-dimensional space. 30

7 The early fusion, mid-fusion, and late fusion (left to right) variants of XceptionNet feature extractor of the RGB+EF XcepTemporal model. 31

8 Visualization of image compression on a test sample. The original frame is shown in the top-left corner. Compression using JPEG Quality 75, 50, and 25 are displayed in the top-right, lower-left, and lower-right corners, respectively. 48

9 Visualization of the attacks on a test sample. The sample is administered with the least and most perturbations of the corresponding attacks exercised in the experiments. 57

List of Tables

1 Data splits for our combined dataset, which is composed of the FaceForensics++ and Celeb-DF datasets. 41

2 **Within-Domain Accuracy:** Frame-level accuracy on the FaceForensics++ [62] and Celeb-DF [44] official test sets. Abbreviations: Face2Face (F2F), FaceSwap (FS), Deepfake (DF), and NeuralTextures (NT). 42

3 **Within-Domain Results:** Video-level accuracy on FaceForensics++ [62] and Celeb-DF [44] official test sets.
Abbreviations: Face2Face (F2F), FaceSwap (FS), Deepfake (DF), and NeuralTextures (NT). 44

4 **Cross-Domain Results:** Models trained on FaceForensics++ [62] and Celeb-DF [44] and then tested on out-of-sample DFDC-preview [17] and YT datasets. Method-A (M-A) and Method-B (M-B) are two types of manipulated videos released in the DFDC-preview dataset. 46

5 **Compression:** Results for different rates of compression. Results in **bold** indicate the best results across the four tested models for the given setting. Note that high JPEG quality means *less* compression, while high MPEG quantization means *more* compression. 47

6 **Compression:** Models trained on augmented training data consisting of the original and JPEG compressed (Quality-50) versions of all the training faces. 49

7 **Transfer learning/Domain Adaptation:** Models are initialized on Face-Forensics++ [62] and trained on Celeb-DF [44] along with 50 out of 720 Faceforensics++ videos of each class to prevent forgetting. 51

8 **Ablation:** Accuracy of XcepTemporal (CE) with different recurrent structures. 52

9 **Ablation:** Accuracy of XcepTemporal (CE) on the combined test sets for various length and stride combinations. 53

10 **Within-Domain Results:** Our models, CNN+Temporal and RGB+EF XcepTemporal, and their variants are trained on the train set and evaluated on the test set of DFDC-mini dataset. 54

11 **Cross-Domain Results:** Our models, CNN+Temporal and RGB+EF XcepTemporal, and their variants are trained on the FaceForensics++ [62] train set and evaluated on the DFDC-mini dataset. 55

12 **Pre-Immunization Results:** Our models, CNN+Temporal and RGB+EF XcepTemporal, and their variants are evaluated on the adversarial attacks administered on the whole DFDC-mini dataset. The accuracies reported are overall accuracy across the test set of the dataset. 58

13 **Post-Immunization Results:** Our models, CNN+Temporal and RGB+EF XcepTemporal, and their variants are evaluated on the adversarial attacks administered on the whole DFDC-mini dataset. The accuracies reported are overall accuracy across the test set of the dataset. 58

Abstract

Deepfakes, or artificially generated audiovisual renderings, can be used to defame a public figure or influence public opinion. With the recent discovery of generative adversarial networks, an attacker using a normal desktop computer fitted with an off-the-shelf graphics processing unit can make renditions realistic enough to easily fool a human observer. Detecting deepfakes is thus becoming vital for reporters, social networks, and the general public. Preliminary research introduced simple, yet surprisingly efficient digital forensic methods for visual deepfake detection. These methods combined convolutional latent representations with bidirectional recurrent structures and entropy-based cost functions. The latent representations for the video are carefully chosen to extract semantically rich information from the recordings. By feeding these into a recurrent framework, we were able to sequentially detect both spatial and temporal signatures of deepfake renditions. The entropy-based cost functions work well in isolation as well as in context with traditional cost functions.

However, re-enactment based forgery is getting harder to detect with newer generation techniques ameliorating on temporal ironing and background stability. As these generative models involve the use of a learnable flow mapping network from the driving video to the target face, we hypothesized that the inclusion of edge maps in addition to dense flow maps near the facial region provides the model with finer details to make an informed classification. Methods were demonstrated on the FaceForensics++, Celeb-DF, and DFDC-mini (custom-made) video datasets, achieving new benchmarks

in all categories. We also perform extensive studies to evaluate on adversaries and demonstrate generalization to new domains, consequently gaining further insight into the effectiveness of the new architectures.

1 Introduction

While misinformation being spread through interpersonal communication and mass media is hardly new, we are at a pivotal point in history where advances in digital communication and AI, in particular, threaten to magnify misinformation's scale, persistence and consequence [20, 57]. Historically, journalists have played a significant role vetting information before publication and dissemination. Now, however, with deceptive information harder to verify, journalists accidentally threaten to contribute to information anarchy, stoking unfounded fears in the public. At best, false information may manipulate emotions and opinions. At worst, it could lead to organized and destabilizing public actions united behind false ideas or impressions.

Deepfakes are artificially generated audiovisual renderings of a person. These recordings, which are typically done without consent, can be used to defame a public figure or influence public opinion. Much like malicious computer viruses, deepfake generation [38, 21, 47, 79, 28, 29, 11, 58, 90, 68, 60, 78, 45, 82, 23, 56, 41, 89, 69, 88, 83, 82] and deepfake detection [1, 70, 22, 55, 62, 14, 43, 54, 13, 4, 40, 18, 6, 42, 7, 16, 52, 85, 10, 2, 3, 87] techniques are continually evolving. As time passes, deepfake generation methods are not only becoming more realistic or believable, but they are learning to better circumvent detection methods. In parallel, deepfake detection methods are learning to recognize subtle hints or fingerprints inadvertently introduced by improved generation methods.

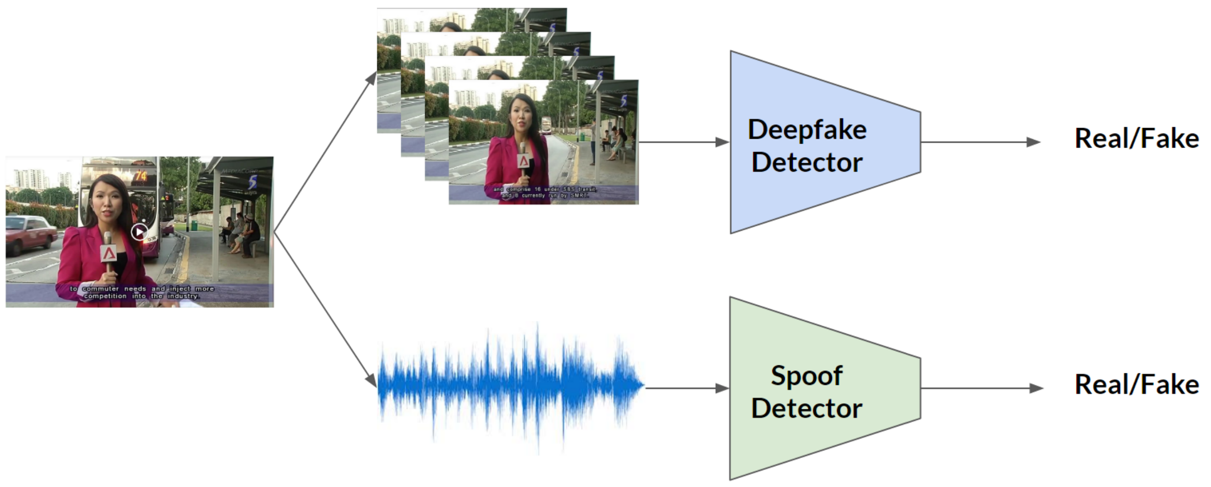

Figure 1: An overview of the problem space. Audio and visual tracks are extracted and processed through spoof and deepfake detection models.

Deepfake creation methods typically rely on face swapping [38, 47] or face synthesis [79] combined with audio dubbing, voice conversion or voice synthesis [46]. Detection methods have concentrated on structure [1, 70, 22, 55, 62, 14, 54] or soft biometrics [43, 13, 4] in visual data, as well as spoof detection for audio [35, 80, 37]. Digital forensic methods can be very good at detecting targeted forgeries, but they struggle at *out-of-sample*, or unforeseen creation methods.

Inspired by the XceptionNet [12] architecture and convolutional recurrent neural network methods [22, 71], we introduce simple, yet effective architectures for the detection of both deepfake video and audio. As shown in Fig. 1, in agreement with current datasets and challenges, we treat audio and video as separate problems. The audio and video streams are referred to as *spoof detection* and *deepfake detection*, respectively. This research work focuses on the deepfake detection stream but not the audio network because there doesn't exist a dataset where both the audio and the video channels are manipulated.

2 Literature Review

2.1 Deepfake Generation

Deepfake creation is a relatively new field in digital forgery. Although the generation process began with traditional vision and voice impersonation, most recent works involve generative adversarial networks (GANs). We will first describe the traditional approaches, and then the adversarial methods.

In general, there are two types of deepfake generation, face replacement and face re-enactment. In face replacement, the face of a target person is overlaid on the face of a source. The overlaid faces are post-processed to blend the edges such that they match the source's facial outline. This can be used, for example, to make Nicholas Cage (the *source* actor) appear in a movie in place of the original (*target*) actor.[1] Faceswaps [39] are a graphical approach, where the facial landmarks (nose, eyes, eyebrows, lips, chin, and cheek areas) play a major role in morphing the target's face with the source's, and the output is post-processed with edge polishing and color correction.

Facial re-enactment, on the other hand, is used to make a target appear to act and speak like the source. This is used, for example in the videos where former U.S. President Barack Obama is made to say whatever the source video says.[2] Facial re-enactment techniques

[1]https://www.theguardian.com/technology/ng-interactive/2019/jun/22/the-rise-of-the-deepfake-and-the-threat-to-democracy

[2]https://www.buzzfeednews.com/article/davidmack/obama-fake-news-jordan-peele-psa-video-buzzfeed

model both the source and target faces to identify facial landmarks and manipulate the target's landmarks to match the source's facial movements. Face2Face [79] is a face re-enactment method that translates the facial expressions of a source subject with a target while maintaining the facial features of the target.

More recent Deepfake methods are based on Generative Adversarial Networks (GANs) [21]. GANs consist of two competing neural networks: a generator G that generates fake samples that mimic real samples from a target dataset, and a discriminator D that tries to tell fakes apart from real samples. These two networks are trained simultaneously, such that over the training period, both the generator and discriminator improve. Upon successful convergence, the generator can then be used to create realistic-looking examples. To promote variation, G is seeded with a noise vector, but this noise vector can be paired with a latent representation of an object (word, image, sentence), to constrain the resulting image [61].

Zhu et al. [91] and Kim et al. [31] built on the GAN concept and replaced the noise vector with input images. This modification enabled their cycle-consistent GANs to alter the domains of the output images based on the input image domains. Similarly, in deepfake generation, it is possible to retain the facial expressions of a source person while transferring identities to a target person. Lu et al. [48] proposed an identity-guided conditional CycleGAN to translate low-resolution face images to high-resolution face images. Kim et al. [30] accomplished a similar transfer, with the difference being that they transfer expressions as well as 3D pose into a target image, creating a video in the process. Faceswap-GAN [47] created realistic looking imagery using adversarial and perceptual losses. The addition of a perceptual loss was shown to minimize unnatural artifacts such as awkward eyeball movements. Temporal smoothing of the frame-to-frame face detection box and an attention mask made the created videos appear more realistic. NeuralTextures [78] is a facial re-enactment forgery that relies directly on facial landmarks. These fakes are generated using a patch-based adversarial loss alongside a photometric reconstruction loss.

Wang et al. [83, 82] proposed flow-based video-to-video synthesis that required training on multiple talking videos of the same person for face re-enactment forgery and revamped the network to only use a single image of a person in generating a deepfake. Siarohin et al. [69] introduced a learnable optical flow network approximating it to a first-order Taylor polynomial. Their method enabled few-shot capability to generate a manipulated video of a person using a single image. Li et al. [41] devised an adaptive-attention-based denormalization generator for high-quality face replacement using a novel learning metric. In an interesting twist, Suwajanakorn et al. [72] achieved photorealistic results by only requiring audio as an input to generate forged videos. Using the weekly addresses of Barack Obama, the raw audio features are first mapped to mouth shapes, then mouth textures, then 3D pose mapping, and finally compositing of the head and torso from stock footage.

2.2 Deepfake Detection

Since the threats posed by deepfake video manipulations became apparent in early 2018, several works have looked into detecting them. A few of the detection techniques targeted handcrafted features [43, 13, 50] like blinking inconsistencies, biological signals, and unrealistic details. Most of these manually crafted detection features exploit known weaknesses in creation methods. Like a cat-and-mouse game, deepfake creation methods quickly adapted to circumvent detection, and the cycle repeats. Recent detection methods [40, 18, 6, 42, 7, 16, 52, 85, 10] rely on machine learning on deepfake datasets to automatically discover forgeries from real videos.

MesoNet [1] uses a shallow convolutional network to detect forgery at a *mesoscopic* (or intermediate) level of detail, intentionally avoiding focusing too much on microscopic features that could be lost due to video compression. They also introduced a variant of their model that replaces regular convolution blocks with *MesoInception* blocks to get slight improvements. The Capsule-Forensics method proposed by Nguyen et al. [55] uses capsule networks [25] for the detection of replay attacks as well as computer-generated images and videos. They argue that the chances of detecting high-quality forgeries would be increased with the agreement between capsules through dynamic routing [65]. Cozzolino et al. [14] applied an autoencoder-based architecture to show its usefulness for transfer learning. Nguyen et al. [54] extended Cozzolino et al.'s network by replacing the standard decoder with a decoder that additionally generates a mask of the manipulated region through multi-task learning.

XceptionNet [12] is one of the more promising deep neural models for feature extraction when trained either from scratch or with pre-trained ImageNet [15] weights. As such, we choose XceptionNet as an embedding space instead of the networks used in prior works [22, 71]. The network architecture not only uses skip connections akin to ResNet [24] with an Inception-like [74] convolutional arrangement, but it also has a modified version of depth-wise separable convolutional layers for reducing the number of parameters with marginally better performance.

While the previously mentioned methods targeted intra-frame inconsistencies, Güera and Delp [22] introduced a spatio-temporal model with InceptionV3 [74] as the feature extraction network to tackle deepfakes. The features from their time-distributed extraction network are forwarded to a unidirectional Long Short-Term Memory (LSTM) network that ultimately makes the classification decision. Sabir et al. [64] evaluated the same architecture with different feature extractors [27, 24]. Their face-extraction process was adjusted by aligning the faces from consecutive frames using facial landmarks to maintain temporal consistency. While their performance on manipulated videos [22] from the HOHA dataset [84] was quite good, performance on the larger and more complex FaceForensics++ [62] dataset was not as effective.

Agarwal et al. [4, 2, 3] took an alternative approach to detection by treating the problem as one of anomalous behavior detection. They learn the typical behavior of the subject from real videos by extracting facial landmarks and temporal actions and use them to train novel networks for behavioral understanding. They also model correlation-based anomaly detection between lip movements and phonemes/visemes. Although these methods cannot be used on unforeseen faces, it is an effective option to preserve robustness against future improvements in other aspects of deepfake generation. For example, they claim there is already sufficient data for most world leaders, although it would be a considerable undertaking to train a detector for all government officials, corporate leaders, or persons of interest.

It is also established that GANs leave their impression on the samples they generate. Yu et al. [87] discuss the potential of GAN fingerprinting analysis by developing a network that is able to extract the model fingerprint from the samples. However, this technique can only be used as a baseline to improvize on because their work can't be generalized for a plethora of deepfake generators.

Although most of the deepfake detection models are accurate on paper, they are unequivocally prone to adversarial attacks. Some of the works [59, 53, 9, 63] analyze the pitfalls of the state-of-the-art detectors by delving into the adversaries and demonstrate the need for robust forgery detectors.

Additionally, recent survey papers [81, 77, 51] in the domain present a comprehensive understanding on the state-of-the-art methods and comparative analysis.

3 Background

3.1 CNNs for feature extraction

Facenet [67] provides a unique embedding space for facial recognition and clustering. This deep convolutional neural network uses triplet loss for training where the objective is to not only minimize the distance between the feature points of the same face in a euclidean space but also maximize the distance between different faces. Triplet loss is defined as

$$\sum_{i}^{N}[||f(x_i^a) - f(x_i^p)||_2^2 - ||f(x_i^a) - f(x_i^n)||_2^2 + \alpha]_+ \tag{1}$$

where a, p, and n are anchor, positive, and negative samples respectively. $f(x)$ represents the embedding vector of an image. α is defined as the margin between positive and negative pairs. It is essentially a threshold that determines how distant positive pairs should be relative to that of negative pairs. A secondary advantage of using triplet loss is that there exists a large set data samples from all the possible combinations of anchors and positives even if the dataset isn't large enough. The embedding space of the Facenet architecture is designed to be 128-dimensional.

XceptionNet (Fig. 2) [12] is a deep convolutional neural network that enhances feature abstraction by using modified depthwise separable convolutions instead of traditional 2D

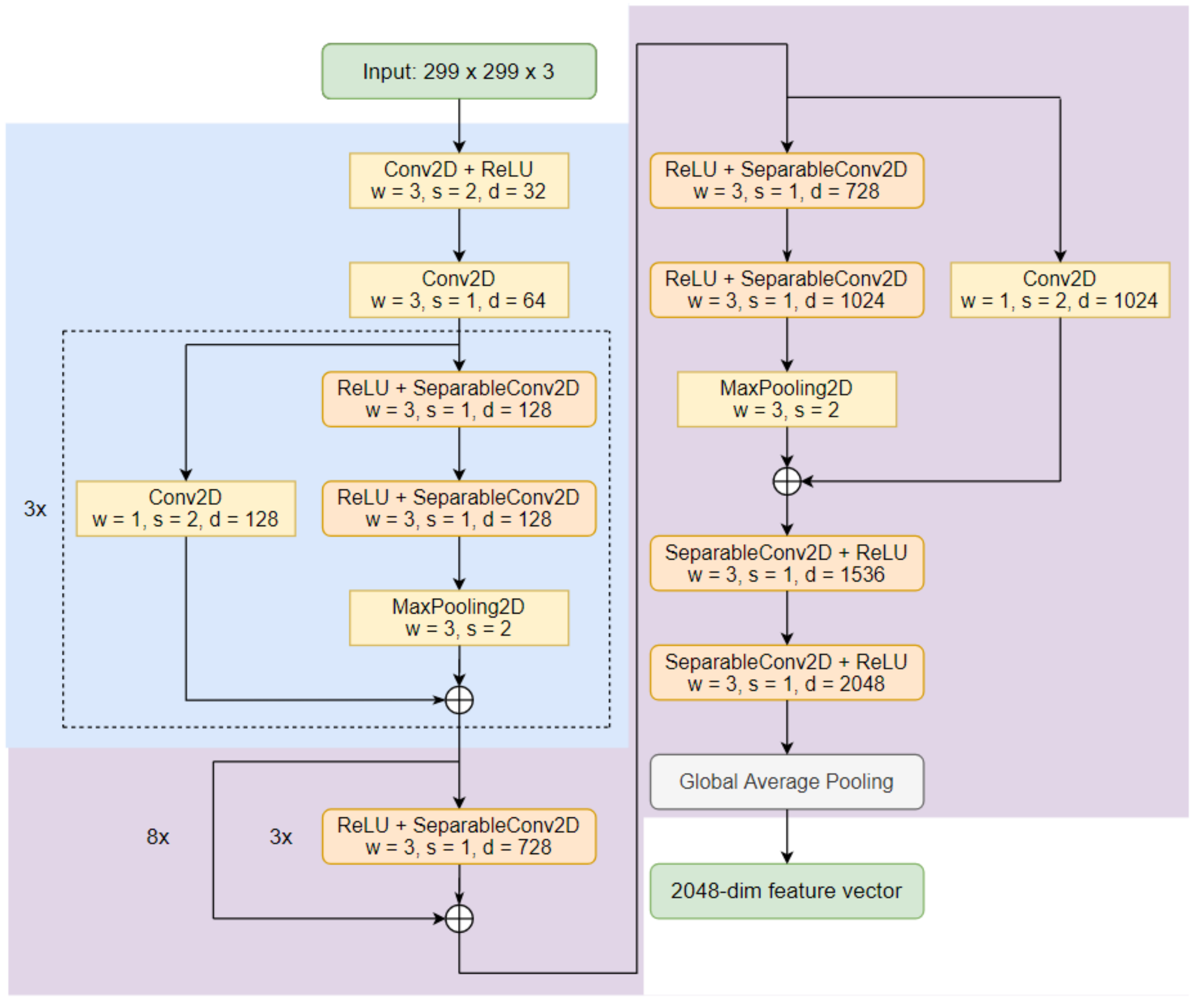

Figure 2: XceptionNet as a feature extraction model, converting each video frame to a latent vector embedding.

convolutions. As the architecture is a deep network, residual connections establish an identity nexus for the gradients to be propagated in a better fashion without any depreciation. The model produces a 2048-dimensional feature space for a 299 x 299 RGB image. On the Imagenet [15] leaderboard, XceptionNet is shown to perform better in a classification setting than traditional deep networks [70, 24, 73].

EfficientNet [76] is a neural architecture designed using neural architecture search (NAS).

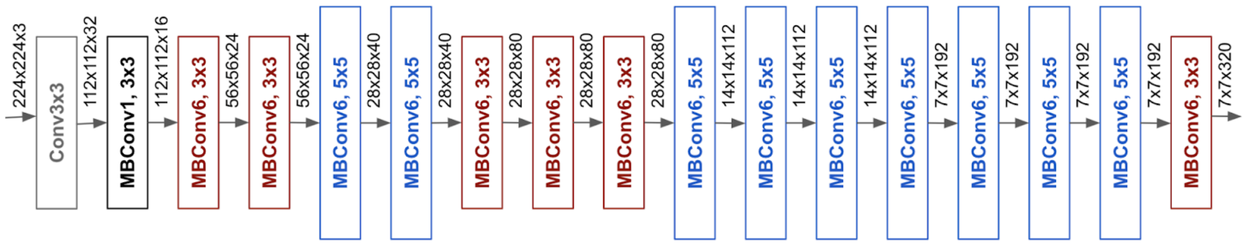

Figure 3: EfficientNet baseline archiecture [76].

NAS is a technique for automating the design process of a deep artificial neural network. Traditionally, CNNs are designed in an austere fashion and scaled up to improve the performance. The conventional method of scaling involves either the width or the depth of the network. The authors present a compound model scaling technique against available resources to improve the performance of a model. They perform a grid search of the available scaling options to determine the scaling coefficients and test the performance of the model layer-by-layer. Using this information, they scale the baseline network to the desired size which is computationally viable as constrained.

However, to design an efficient model as a whole, the baseline architecture should be able to extract adequate features from the image dynamics. This hypothesis drove the team to rely on MnasNet [75] for the design of the baseline network. They scaled up the baseline to obtain a family of models, named EfficientNets (variants ranging from B0 to B7). In comparison with the traditional classifiers [70, 24, 73, 12] on Imagenet [15], EfficientNets showed outstanding performance with a reduction in parameter size and floating point operations. Likewise, they also proved to be consistent in transfer learning applications.

3.2 Long Short-Term Memory (LSTM)

Recurrent units from RNNs suffer from perpetuating short-term memory. If a set of features are passed into recurrent units in multiple timesteps, RNNs fail in utilizing features from earlier timesteps to later timesteps when making an informed decision pertaining to the task. LSTMs were created as a solution to the short-term memory drawback. They are designed with internal gates that regulate the flow of information between timesteps.

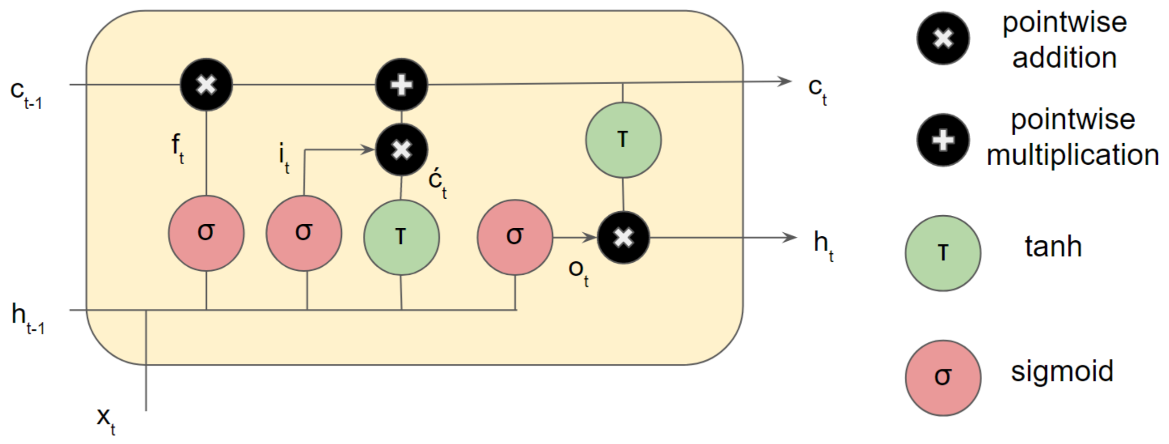

Figure 4: An LSTM unit (adapted from [26])

The following equations describe the computations involved in an LSTM cell -

$$
\begin{aligned}
i_t &= \sigma(x_t U^i + h_{t-1} W^i) \\
f_t &= \sigma(x_t U^f + h_{t-1} W^f) \\
o_t &= \sigma(x_t U^o + h_{t-1} W^o) \\
\acute{c}_t &= tanh(x_t U^g + h_{t-1} W^g) \\
c_t &= \sigma(f_t * c_{t-1} + i_t * \acute{c}_t) \\
h_t &= tanh(c_t) * o_t
\end{aligned}
\tag{2}
$$

where i, f, and o indicate input, forget and output gates respectively. Additionally, c stands for the cell state and h stands for the hidden state of an LSTM unit.

The core idea of a working LSTM is its cell state and the operations carried out by the gates. In theory, the cell state is expected to carry relevant information down the temporal sequence of the recurrent layer. During training, the gates act as tiny neural units that decide which information is considered relevant. Information from the previous hidden state and the current input is passed through the forget gate f_t where the sigmoid activation decides if the information needs to be memorized or ignored. At the input gate i_t, $tanh$ activation constrains the information between -1 and 1, thus regulating the network. This knowledge is multiplied with the resultant forget state to ensure that only relevant information is propagated. Eventually, the output gate o_t decides what the next hidden state h_t should be. The hidden state is responsible in carrying the information from the previous inputs of the LSTM layer.

3.3 Optical Flow

Naively, optical flow is defined as the pattern of motion caused by moving objects in a video. OpenCV [8] provides extracting two types of flow from a visual motion sequence, namely sparse flow and dense flow. Sparse flow uses Lucas-Kanade [49] method and produces the flow vectors of salient feature set of pixels in an image i.e., corners or geometric center of objects whereas dense flow generates instance-level flow vectors of the entire frame regardless of the object in focus. For dense flow, OpenCV computes the magnitude and direction of the flow from a 2-D array of flow vectors. OpenCV's dense flow uses Gunnar Farneback's [19] algorithm.

3.4 Edge Detection

Edge detection involves a variety of mathematical transformations on the image that aim to extract the edges at the pixels across which there are discontinuities in the intensity. Sobel, Prewitt, Laplacian, and Canny are some of the most common edge detection algorithms. Sobel, Prewitt, and Laplacian edge detectors use simple two-dimensional filters that operate on the image to produce an edge map whereas Canny edge detection is relatively complex in functioning.

Canny edge detector is also the most effective of the aforementioned algorithms. It follows a series of operations on an image in order to produce an edge map. The first step of Canny edge detection involves applying Gaussian filter on the image to attenuate the noise. Now, the gradients (and their orientation) of the filtered image are computed using either of Sobel or Prewitt edge detectors. To this intermediate map, non-maximum suppression is applied so that remaining noisy pixels and thin broken edges are suppressed. This leaves out both properly drawn edges and broken edges due to suppression applied in the previous step. Hysteresis is a method of linking such broken edges that are meant to be connected. Using appropriate thresholding, smaller edges can be ignored and perceptible ones are conjoined.

4 Proposed Deepfake Detection Architecture

Although newer deepfake generative algorithms process multiple frames at a time and devise a flow-based learning metric, the existence of temporal inconsistencies is plausible regardless of any post-processing during their generation. To equip ourselves in detecting any forged audio or video, we discuss the advantages of recurrent conformations over convolutional feature extractors by presenting deep neural models in visual domain. Our architecture for deepfake detection is inspired by the spatio-temporal processing used in prior works [22, 71].

We use a convolutional architecture to obtain a vector representation of a facial region of a frame, f_i. A sequence of such facial regions from frames, $f_1, f_2, \ldots, f_f$ are passed into a bidirectional LSTM module to learn a latent representation capable of discriminating between facial manipulations and original faces. Additionally, we leverage the optical flow from the video to endow the model with dense visual features and facilitate in robust discrimination of deepfakes apart from the original videos.

4.1 Preprocessing

4.1.1 Face Extraction

The dlib [32] face detector determines the primary face over each frame in the video. Canonical face images are generated by cropping to the dlib facial bounding box with a multiplier three times the area of the extracted bounding box, resampling to 299×299 pixels, and normalized to zero mean and unit variance.

As dlib extracts faces individually for each frame, we observed that the subsequent face images have subtle differences in the box coordinates. Similar to Faceswap-GAN [47], we apply a linear smoothing filter over the box coordinates of consecutive frames to mitigate the temporal inconsistency introduced in the extraction process.

4.1.2 Optical flow and Edges

We use OpenCV [8] to extract the dense optical flow from the consecutive frames in a video. If there exist n frames in a video, we get $n-1$ flow maps. The dense optical flow is generated using three layers of pyramids where each layer is half the size of the previous. The window size for constructing the flow is set to 15 for three iterations. Then, the flow is represented in RGB channels to signify the magnitude and direction of motion.

Figure 5: The EdgeFlow (EF) input shown on the right is generated using the test sample on the left.

In addition to the flow map between k^{th} and $(k+1)^{th}$ frames, we propose supplementing the dense flow map with the edges produced in the k^{th} frame. OpenCV's canny edge detection is employed for this strategy. The OpenCV function for canny edge detection has a provision for manual threshold apart from the image. The threshold is set to 100 below which all edges detected are discarded. The canny edge function returns a grayscale image with sparse features which is converted into an RGB image.

Eventually, an edgeflow map is devised by element-wise addition between the edge map and the flow map; as depicted in Figure 5.

4.2 Architectures

4.2.1 CNN+Temporal Model

Canonical faces are encoded using the CNNs described in Section 3.1. We propose FacenetLSTM, XcepTemporal, and EffTemporal; convolutional recurrent networks with multiple levels of temporal feature abstraction. The spatial features from the CNN module are passed into a first bidirectional-LSTM layer. The outputs of the first bidirectional-LSTM layer are passed to a second bidirectional-LSTM layer to produce secondary feature abstraction. The feature vector from the last LSTM unit of this second bidirectional layer is passed into a fully-connected layer and finally to a classification layer. Dropout is added to the fully-connected layer for regularization.

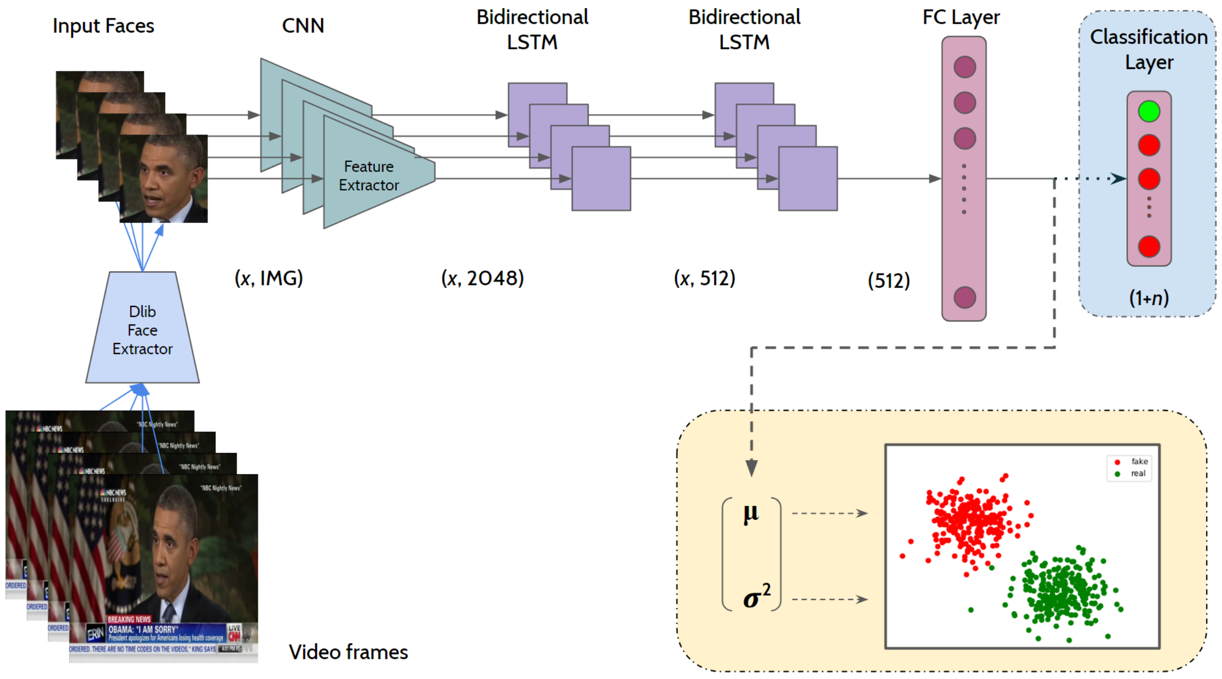

Figure 6: The CNN+Temporal model. Faces are first extracted and normalized to a canonical representation. These canonical images are passed into a convolutional recurrent model to make a prediction. μ and σ^2 represent the mean and variance of the distribution in a two-dimensional space.

4.2.2 RGB+EF XcepTemporal Model

Although the CNN+Temporal model seemed to work well for within-dataset test samples, it is still only an initial step towards building robust deepfake detectors. The CNN+Temporal model is sensitive to visual constituents like lighting, blur, noise, and compression. This makes the model vulnerable to manipulations by such adversarial elements.

We propose leveraging the dense motion facets of the video as it not only invigorates resilience against adversarial attacks but also provides the recurrent layers with beyond-the-visual flow attributes which could ultimately be used in making a better decision.

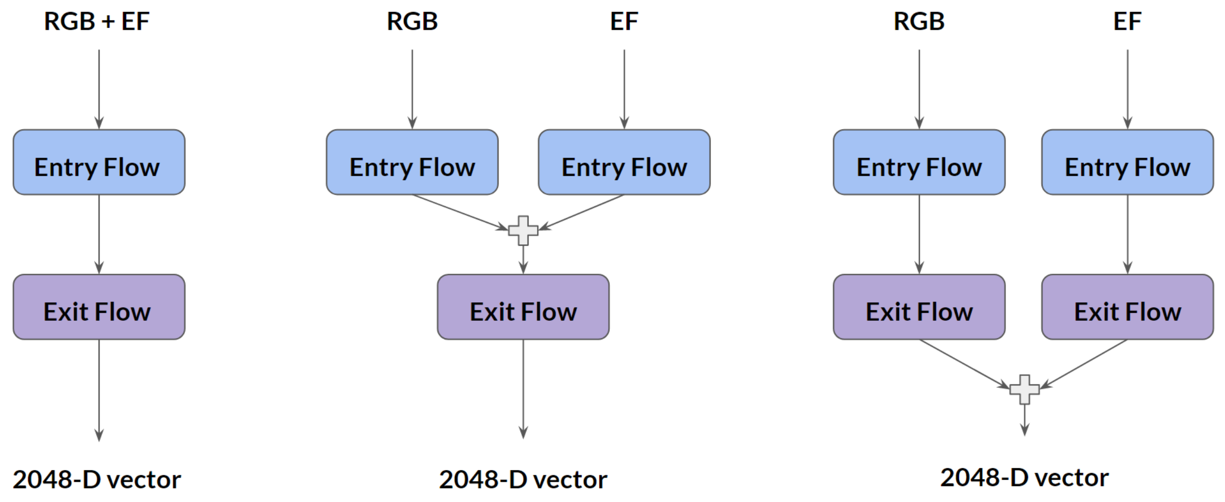

Figure 7: The early fusion, mid-fusion, and late fusion (left to right) variants of XceptionNet feature extractor of the RGB+EF XcepTemporal model.

As described in Section 4.1.2, we devise edge flow maps that can be used to enhance the feature abstraction ability of the CNN feature extractor. We chose XceptionNet [12] as the feature extractor in designing the RGB+EF variants of the model. The entry flow and exit flow of XceptionNet described in Figure 7 are shown in blue and purple regions in Figure 2 respectively. The *plus* (+) symbol indicates element-wise addition of features at that particular layer.

4.2.3 Loss Functions

Faces from real videos are hypothesized to have their own embedding distribution, while different types of generated fake videos can either be clustered together or in disparate distributions. The main goal in our system is to discriminate the real video distribution from that of the forged videos. To this end, we compare the use of two loss functions for learning the discrimination accurately: Cross-entropy and Kullback-Leibler (KL) divergence.

Cross-entropy loss is the conventional loss used in classification and is defined as

$$L_{CE} = -\log\left(\frac{e^{y_c}}{\sum_{j=1}^{1+n_f} e^{y_j}}\right) \tag{3}$$

where y_c is the ground truth logit value and $(1 + n_f)$ represents one class for real videos alongside n_f deepfake classes.

KL divergence, also known as relative entropy, is a natural approach to measure the differences amongst probability distributions. Inspired from the application of KL divergence in variational autoencoders [34], the high-level idea is to disentangle different probability distributions via parameter learning. We hypothesize that the probability distribution of the real and ersatz material [5] when mapped into a two-dimensional space from the latent feature representation generated after the primary fully-connected layer can be disentangled.

In this setting, mean (μ) and variance (σ) of the bivariate normal distribution $\mathcal{N}$ are estimated to be computed from the latent feature vector. The loss function encourages the model to distinguish one distribution from another. The bivariate KL divergence is represented as:

$$D_{KL}(\mathcal{N}\,((\mu_1, \mu_2)^T, diag(\sigma_1^2, \sigma_2^2))\,||\,\mathcal{N}\,(0, I)) = \lambda\ (\sum_{i=1}^{n} \sigma_i^2 + \mu_i^2 - log(\sigma_i) - 1) \tag{4}$$

We use $\lambda = (1/2)$.

Similar to the Fisher linear discriminant loss, the KL loss is equivalent to the sum of class-wise KL divergences. Intuitively, the KL loss distributes all the learned samples of each class in a densely packed normal distribution and clusters the samples from one class further away from other classes. The loss composition of the KL-divergence metric is defined as:

$$L_{KL} = \sum_{i=1}^{n} y_i \, log(\frac{y_i}{\hat{y}_i}) \tag{5}$$

The choice of using a two-dimensional space was determined empirically.

We also propose to use an ensemble of the learning procedures with a convex combination of the cross-entropy and KL divergence losses. The ensemble loss function is defined as:

$$L_{EN} = \lambda_1 \, L_{KL} + \lambda_2 \, L_{CE} \tag{6}$$

where λ_1 and λ_2 are the weights allocated to respective loss metrics to provide a flexible overall loss function. We typically allocate a larger weight to KL divergence than cross-entropy for deepfake detection, as KL divergence aids in driving the real and fake distributions further apart.

4.2.4 Variants

As we proposed to make use of two different loss metrics, they were made use in isolation and also in combination to define the variants of our CNN+Temporal model. We then choose the best performing loss metric across the variants of the CNN+Temporal network in RGB+EF XcepTemporal model. Within the XceptionNet model architecture of RGB+EF XcepTemporal model, we introduce various fusion techniques and comprehend which of them performs the best during inference.

We propose four variants of the CNN+Temporal model:

1. CNN+Temporal (CE).
   - CNN+Temporal with cross-entropy loss L_{CE}.
   - It follows the horizontal line path to the blue box labelled "Class Layer" in Figure 6.
2. CNN+Temporal (KL).
   - CNN+Temporal with KL divergence loss L_{KL}.

- It follows the dashed line path to the yellow box labelled "KL-Divergence" in Figure 6.

3. CNN+Temporal (EN).

   - CNN+Temporal with ensemble loss L_{EN}.
   - This variant has two classes in the classification layer of the model (real and fake).
   - It has a Y-shaped final layer where the outputs are a two-class classification layer and a sample in two-dimensional space.

4. CNN+Temporal (EN$_{1+n}$).

   - CNN+Temporal with ensemble loss L_{EN}.
   - This variant has $1 + n_f$ classes in the classification layer of the model (one real and n_f fakes).
   - It has a Y-shaped ultimate layer where the outputs are a $(1 + n_f)$ - class classification layer and a sample in two-dimensional space.

- This variant not only distinguishes the distribution of fakes from original faces but also determines the type of fake.

We propose three variants of the RGB+EF XcepTemporal model:

1. RGB+EF XcepTemporal (Early Fusion).

    - RGB+EF XcepTemporal with KL divergence loss L_{KL}.
    - The CNN feature extraction is depicted on the left in Figure 7.
    - Both the inputs i.e., the face and its edge flow are fused early to produce a 6-channel input for the XceptionNet model

2. RGB+EF XcepTemporal (Mid-Fusion).

    - RGB+EF XcepTemporal with KL divergence loss L_{KL}.
    - The CNN feature extraction is depicted in the centre of Figure 7.
    - Both the inputs have separate streams of entry flow and are element-wise added to have a single exit flow.

3. RGB+EF XcepTemporal (Late Fusion).

    - RGB+EF XcepTemporal with KL divergence loss L_{KL}.
    - The CNN feature extraction is depicted on the right in Figure 7.
    - Both the inputs have separate streams of entry and exit flows and their feature vectors are element-wise added to produce a single feature vector.

5 Results

5.1 Datasets

UADFV [86] uses traditional vision approaches to generate manipulated videos. The generation process maps the facial landmark points of the source to that of the target. This dataset consists of 49 real and 49 fake videos.

DeepfakeTIMIT [36] was built from the VidTIMIT [66] dataset using a GAN-based face swap [47] over dataset subjects. This dataset uses 32 subjects, each with 10 face swap videos of both low and high quality for a total of 620 fake videos.

FaceForensics++ [62] contains four different types of deepfakes, namely Face2Face [79], FaceSwap [47], Deepfakes, and NeuralTextures [78], alongside corresponding real videos. The dataset contains 1000 real source videos and 1000 of each of the four deepfake generation methods. Since the traditional vision and face swap methods used to create the UADFV and DeepfakeTIMIT datasets, respectively, are also in FaceForensics++, most works do not report on UADFV or DeepfakeTIMIT.

The FaceForensics++ dataset was updated on 23rd August 2019 to host the **Deep Fake Detection (DFD)** dataset from Google and JigSaw [62], which consists of 360 original source videos and 3000 manipulated videos.

Celeb-DF [44] was built from 400 original YouTube videos and 800 synthesized videos. Unlike the aforementioned datasets, these fakes are refined to address issues with color inconsistency, low-frequency smoothing, and temporal flickering. Additional refinements include synthesis of higher-resolution fakes (256×256), whereas the algorithms used in previous works synthesized low-resolution (64×64) fakes.

Deepfake Detection Challenge (DFDC) [17] by Facebook consists of 19k real videos and 95k deepfakes generated using the real videos. The manipulated set of the challenge consists of face replacements and video tampering. As our work doesn't focus on detecting visual tampering, a partial dataset, referred to as **DFDC-mini** in the later sections, was isolated with 872 real videos and their corresponding 1241 face replacement videos.

YouTube (YT) consists of 20 real and 20 fake videos that we collected from YouTube.com as a way to test accuracy on unseen samples. The 20 real videos all feature a single camera-facing subject, while the 20 fake videos are from the Ctrl Shift Face YouTube channel.[3] We do not know which deepfake technique the creator of this channel uses, however we do know that he uses face-swapping as opposed to facial re-enactment methods.

The datasets UADFV and DeepfakeTIMIT are not used in this research as the quality of the fakes in these video datasets are visibly distinguishable and prior works already proved

[3] `https://www.youtube.com/channel/UCKpH0CKltc73e4wh0_pgL3g/`

to detect these fakes quite easily. We also chose not to utilize DFD dataset in our work because the samples resemble FaceSwaps from FaceForensics++. We decide to concentrate on FaceForensics++, Celeb-DF, and the more difficult DFDC-mini dataset for evaluating all of our models. To test on the real-world deepfakes, we use DFDC-preview and YT datasets to assess the efficiency of our models.

5.2 CNN+Temporal Model

5.2.1 Baselines

We compare our work with six high-performing deepfake detection models proposed in prior work: CapsuleForensics [55], ClassNSeg [54], ConvLSTM [22], FaceNetLSTM [71], DenseNetAligned [64], and XceptionNet [12]. We use code provided by the authors for this purpose. The performance of most of these models have been previously reported on the same datasets [62, 44]. For a fair comparison, however, we tested them along with our variants of XcepTemporal on identical train and test splits.

Prior work did not describe how to address making a detection decision on an entire video instead of a single frame. As we are interested in results on the entire video and to provide a fair set of baselines for comparison, we propose a simple mechanism for converting a *frame-level* model into a *video-level* model. We first pass the frame-level results, which produce a higher value for frames that more likely to be fake, through a median filter with a window size of five, which helps to reduce false positives by smoothing out the resulting output. We then take the maximum output from these windows as the overall result for the video. While this can result in some false positives, it is important to detect a video as fake when even just a short portion of the video is fake.

We applied this procedure to all of the baseline frame-level models to produce video-level models, and we report the results for both frame-level and video-level models for multiple methods of comparison.

5.2.2 Training

Split	Train	Validation	Test
Videos	4520	904	819
Facial frames	1,437,681	281,462	248,623

Table 1: Data splits for our combined dataset, which is composed of the FaceForensics++ and Celeb-DF datasets.

We trained our CNN+Temporal model and the baseline models on the combination of the full FaceForensics++ [62] and Celeb-DF [44] datasets. For the training, validation, and test splits, we used the instructions provided along with the datasets. As Celeb-DF does not offer any validation split, we randomly chose 50 real and 134 fake videos from the training data to create a validation split. The test sets for both datasets was left unaltered. The splits used are shown in Table 1.

We set the learning rate to 1e-4 with a decay factor of 1e-5. The optimizer is Adam, with β_1

Model \ Dataset	FaceForensics++ [62]						Celeb-DF [44]		
	Real	F2F	NT	DF	FS	Overall	Real	Fake	Overall
ClassNSeg [54]	40.52	76.37	74.78	93.78	69.14	79.76	42.62	49.17	46.31
ConvLSTM [22]	83.13	88.72	85.71	92.91	90.61	87.84	74.76	80.22	78.17
DenseNetAligned [64]	90.68	89.32	87.30	94.05	91.32	90.53	71.66	87.90	81.39
CapsuleForensics [55]	90.20	96.88	97.40	96.00	97.22	95.54	-	-	-
XceptionNet [12]	91.82	96.13	98.14	99.21	99.89	97.03	79.62	95.12	89.55
Our Models									
FaceNetLSTM [71]	94.01	87.20	86.89	89.16	90.60	89.57	72.95	84.11	79.83
XceptionNet (KL)	**100**	**100**	**100**	**100**	**100**	**100**	93.72	98.25	96.89
XcepTemporal (CE)	**100**	98.57	**100**	**100**	**100**	99.71	94.66	98.81	97.01
XcepTemporal (KL)	**100**	**100**	**100**	**100**	**100**	**100**	94.04	99.72	97.73
EffTemporal-B4 (CE)	**100**	**100**	**100**	**100**	**100**	**100**	95.13	98.64	97.21
EffTemporal-B4 (KL)	**100**	**100**	**100**	**100**	**100**	**100**	**95.40**	99.36	**98.03**
XcepTemporal (EN)	87.92	98.22	98.76	**100**	**100**	96.98	92.91	**100**	97.34
XcepTemporal (EN_{1+n})	93.10	96.61	95.02	99.93	99.82	96.89	95.01	99.22	97.83

Table 2: **Within-Domain Accuracy:** Frame-level accuracy on the FaceForensics++ [62] and Celeb-DF [44] official test sets. Abbreviations: Face2Face (F2F), FaceSwap (FS), Deepfake (DF), and NeuralTextures (NT).

set to 0.9 and β_2 set to 0.999, these being the default values suggested by Kingma and Ba's original paper on Adam [33]. A dropout of 0.5 is added to the first fully-connected layer. Based upon hyperparameter tuning results, we set the sequence length to eight frames with a stride of eight. All the variants of the model were trained end-to-end.

5.2.3 Within-Domain Results

The most basic test for deepfake detection is to train and test on the same datasets. We show accuracy based on evaluating one frame at a time in Table 2, and show accuracy across the entire video in Table 3. We find that XceptionNet and EfficientNet based models are very effective almost uniformly across all samples. In particular, the XceptionNet (KL) and XcepTemporal (KL), EffTemporal (CE), and EffTemporal (KL) models achieve 100% accuracy for the entire FaceForensics++ dataset for both frame-level and video-level detection, while XcepTemporal (CE) is close behind at 99.71% for frame-level detection and 100% for video-level detection.

Among prior works, only XceptionNet [12] and CapsuleForensics [55] could achieve 97% or above, where XceptionNet particularly suffers from false positives at over 8% FPR in frame-level detection and over 11% FPR in video-level detection.

The two ensemble methods do not fare as well in FaceForensics++ at about 97% accuracy on frames and 99% accuracy on full videos, but they reach new standards for accuracy on the Celeb-DF dataset. EffTemporal (KL) has the overall best accuracy on frames at 98.03% and 99.53% on full videos.

In our experience with training the models, using the 1.4M facial frames in the training set (see Table 1), the CNN+Temporal (CE) takes four epochs to converge on the training data, whereas the KL variant takes only two to three epochs to converge. The models with L_{KL} loss tend to converge faster than the cross-entropy loss L_{CE} because the latent distribution generated by the model is distinct between the original and the forged faces. By contrast,

Model \ Dataset	FaceForensics++ [62]						Celeb-DF [44]		
	Real	F2F	NT	DF	FS	Overall	Real	Fake	Overall
ClassNSeg [54]	47.14	65.71	60.00	64.29	48.57	57.14	-	-	-
ConvLSTM [22]	90.71	93.57	91.43	94.29	93.57	92.71	71.05	83.95	79.83
DenseNetAligned [64]	89.28	95.00	93.57	96.43	95.71	94.00	73.68	88.88	84.03
XceptionNet [12]	88.57	96.43	98.57	**100**	**100**	96.71	68.42	**100**	89.91
Our Models									
FaceNetLSTM [71]	94.28	92.14	90.71	95.71	95.71	93.71	77.77	86.41	84.03
XceptionNet (KL)	**100**	**100**	**100**	**100**	**100**	**100**	89.48	93.83	92.44
XcepTemporal (CE)	**100**	**100**	**100**	**100**	**100**	**100**	97.37	**100**	99.16
XcepTemporal (KL)	**100**	**100**	**100**	**100**	**100**	**100**	97.37	**100**	99.16
EffTemporal-B4 (CE)	**100**	**100**	**100**	**100**	**100**	**100**	97.64	**100**	99.24
EffTemporal-B4 (KL)	**100**	**100**	**100**	**100**	**100**	**100**	**98.12**	**100**	**99.53**
XcepTemporal (EN)	94.29	**100**	**100**	**100**	**100**	99.14	94.74	**100**	98.32
XcepTemporal (EN$_{1+n}$)	97.86	99.29	97.86	**100**	**100**	99.00	97.37	**100**	99.16

Table 3: **Within-Domain Results:** Video-level accuracy on FaceForensics++ [62] and Celeb-DF [44] official test sets.
Abbreviations: Face2Face (F2F), FaceSwap (FS), Deepfake (DF), and NeuralTextures (NT).

EN variants of the model take the longest to train (ten epochs) because the gradients from both the losses compete with each other. The convergence is slower in the early stages of the training, but once both the channels start learning, the rate of convergence nearly doubles.

The next sections will explore these models further to help understand how the architectural variations affect performance.

5.2.4 Cross-Domain Results

A critical property of deepfake detection models for real-world use is good inference performance on deepfake types not included in the training dataset. One way to measure this performance is to train on publicly available deepfake datasets from Table 1, and test on the unforeseen cross-domain samples in the Deepfake Detection Challenge preview dataset [17]. As shown in Table 4, we use binary class models, such that we expect the unforeseen fakes, M-A and M-B to be both classified as fakes. Table 4 shows that the KL variant of our XcepTemporal model performs significantly better than the CE variant. We believe this is because the KL loss naturally encourages larger margins between the two different distributions. By encouraging existing classes (real vs. forgery) to be far apart, unforeseen cross-domain variations are more likely to fall on the correct side of the boundary.

5.2.5 Compression

A simple but effective way to bypass deepfake detection is to apply compression to the deepfake video. This section examines the performance of our models on two different types of compression techniques (JPEG and MPEG).

JPEG Compression. OpenCV [8] provides an option to compress individual frames with a reduction in quality as shown in Fig. 8. The quality factor is a measure of the compression rate of JPEG images, ranging between 0 and 100. High numbers between 90-100 represent

[3] `https://www.youtube.com/channel/UCKpH0CKltc73e4wh0_pgL3g/`

Model \ Dataset	DFDC [17]			YT	
	Real	M-A	M-B	Real	Fake
ClassNSeg [54]	25.02	**76.21**	71.07	49.00	54.62
ConvLSTM [22]	59.41	20.68	51.39	77.90	92.45
FacenetLSTM [71]	58.32	13.68	69.02	81.38	90.36
DenseNetAligned [64]	69.12	35.17	52.31	75.25	92.30
XceptionNet [12]	54.67	28.63	74.55	93.82	95.25
Our Models					
Xception (KL)	67.18	55.91	76.11	98.35	**100**
XcepTemporal (CE)	87.34	65.99	85.03	**100**	**100**
XcepTemporal (KL)	92.19	73.21	88.34	**100**	**100**
XcepTemporal (EN)	**93.26**	76.08	**90.10**	**100**	**100**

Table 4: **Cross-Domain Results:** Models trained on FaceForensics++ [62] and Celeb-DF [44] and then tested on out-of-sample DFDC-preview [17] and YT datasets. Method-A (M-A) and Method-B (M-B) are two types of manipulated videos released in the DFDC-preview dataset.

minimal visual quality loss, while numbers less than 60 represent visually significant quality loss. Lower quality factor images take up less disk space. Table 5 shows the test accuracies for our models as well as for the DenseNetAligned and XceptionNet models, two of the best models in our results for uncompressed video. Our models perform the best on most settings in the Face2Face, Deepfakes, FaceSwap, and Celeb-DF datasets, while XceptionNet performs the best for JPEG compression on NeuralTextures fakes. We note a severe degradation in deepfake detection performance across all the models for highly compressed faces. The accuracies are significantly more affected by compression on re-enactment-type fakes (Face2Face and NeuralTextures) than the replacement-type (Deepfakes and FaceSwap).

Model	F2F	NT	DF	FS	Celeb-DF
DenseNetAligned [64]					
JPEG Quality = 75	64.69	70.10	76.49	70.33	48.67
JPEG Quality = 50	29.17	25.34	56.60	50.44	13.16
JPEG Quality = 25	0.32	0.26	12.90	18.31	4.03
MPEG Quant. = 20	73.68	62.44	79.51	77.63	-
MPEG Quant. = 40	18.17	20.31	52.11	45.67	-
XceptionNet [12]					
JPEG Quality = 75	75.60	**70.17**	89.32	91.59	63.02
JPEG Quality = 50	50.62	**45.48**	73.38	72.73	30.78
JPEG Quality = 25	**0.81**	**0.53**	20.17	16.84	**7.29**
MPEG Quant. = 20	80.12	75.24	84.25	82.38	-
MPEG Quant. = 40	27.16	25.41	60.05	56.59	-
Our Models					
XcepTemporal (CE)					
JPEG Quality = 75	80.62	69.54	99.40	95.63	82.65
JPEG Quality = 50	42.31	37.06	89.23	80.86	**52.31**
JPEG Quality = 25	0.07	0.03	**37.28**	46.04	1.32
MPEG Quant. = 20	**98.66**	86.16	90.13	88.62	-
MPEG Quant. = 40	**48.75**	16.96	**78.95**	**64.64**	-
XcepTemporal (KL)					
JPEG Quality = 75	**84.33**	61.07	**99.82**	**97.81**	**84.19**
JPEG Quality = 50	**53.05**	18.44	**97.95**	**94.45**	48.68
JPEG Quality = 25	0.02	0.01	18.97	**46.67**	0.81
MPEG Quant. = 20	93.86	**88.81**	**95.02**	**91.17**	-
MPEG Quant. = 40	0.07	**46.04**	37.28	0.03	-

Table 5: **Compression:** Results for different rates of compression. Results in **bold** indicate the best results across the four tested models for the given setting. Note that high JPEG quality means *less* compression, while high MPEG quantization means *more* compression.

MPEG Compression. The models were additionally tested on the effects of video compression using videos saved at two different H.264 quantization factor levels, as previously

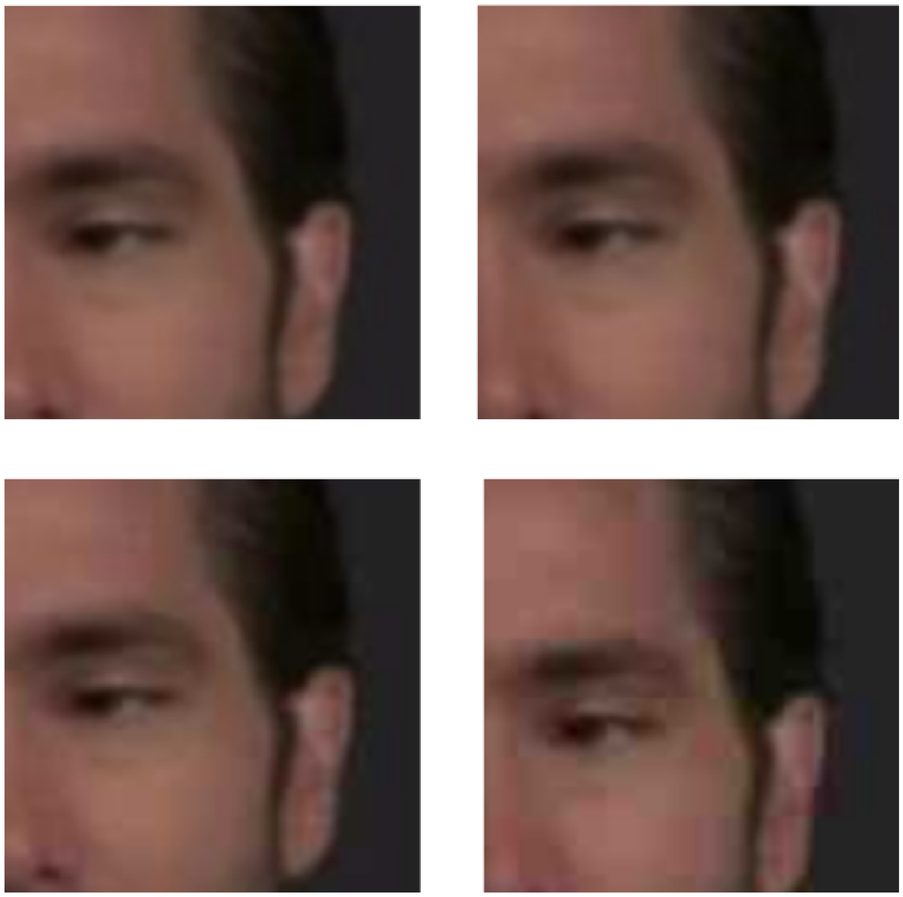

Figure 8: Visualization of image compression on a test sample. The original frame is shown in the top-left corner. Compression using JPEG Quality 75, 50, and 25 are displayed in the top-right, lower-left, and lower-right corners, respectively.

explored by Agarwal et al. [4]. The quantization factor reflects the MPEG compression rate of the videos. The higher the quantization factor, the lower the quality and greater the compression. We selected a quantization value of 20 to represent high-quality MPEG videos and 40 to represent low quality MPEG videos. Table 5 shows a drastic drop in accuracy at the quantization factor of 40.

Model	F2F	NT	DF	FS	Celeb-DF
DenseNetAligned [64]					
Original	86.16	84.61	87.16	83.01	81.62
JPEG Quality = 75	85.16	82.06	87.81	85.02	82.57
JPEG Quality = 50	83.81	80.04	75.28	78.93	81.36
JPEG Quality = 25	71.68	68.12	66.52	69.30	72.61
MPEG Quant. = 20	83.18	82.90	86.47	85.12	-
MPEG Quant. = 40	68.23	63.02	59.31	58.80	-
XceptionNet [12]					
Original	91.57	**93.72**	90.68	94.33	92.03
JPEG Quality = 75	90.42	89.56	88.30	89.16	90.91
JPEG Quality = 50	86.88	85.72	82.75	86.31	88.42
JPEG Quality = 25	80.43	79.64	80.11	72.63	85.80
MPEG Quant. = 20	89.14	90.61	90.12	**92.67**	-
MPEG Quant. = 40	88.00	84.56	81.40	**82.52**	-
Our Models					
XcepTemporal (CE)					
Original	**98.55**	93.17	**97.84**	**99.44**	99.8
JPEG Quality = 75	97.9	**93.32**	**98.44**	99.18	99.95
JPEG Quality = 50	97.5	**93.04**	**99.78**	99.2	**100**
JPEG Quality = 25	94.11	81.52	96.47	98.1	98.35
MPEG Quant. = 20	**97.54**	**96.78**	92.05	85.31	-
MPEG Quant. = 40	87.81	**89.84**	**83.42**	68.91	-
XcepTemporal (KL)					
Original	94.26	84.8	97.73	98.68	**99.99**
JPEG Quality = 75	**98.62**	92.66	98.11	**99.8**	**100**
JPEG Quality = 50	**98.01**	92.09	99.15	**99.71**	**100**
JPEG Quality = 25	**94.26**	**84.8**	**97.72**	**98.68**	**100**
MPEG Quant. = 20	97.22	95.03	**94.67**	89.79	-
MPEG Quant. = 40	**90.62**	86.32	78.3	70.01	-

Table 6: **Compression:** Models trained on augmented training data consisting of the original and JPEG compressed (Quality-50) versions of all the training faces.

Data Augmentation. As the compression artifacts divulge an area of weakness for all methods, we considered retraining models with data augmentation. The models from Table 5 are retrained using both the original training data as well as with their corresponding JPEG compressed faces with a quality factor of 50. The test results are presented in Table 6. With a minimal loss in accuracy on the original test sets, all the models gained robustness in detecting forged faces which are compressed at various compression levels. In this setting, our models become the top performers in all but three of the 28 configurations tested. Impressively, the XcepTemporal (KL) model attains 84.8-100% accuracy for JPEG Quality of 25, which has a substantial reduction in visual quality, as shown in Figure 8. In contrast, neither DenseNetAligned nor XceptionNet got above 81% in any of the tests with this low quality. On MPEG compression, our models also perform best in the Face2Face, NeuralTextures, and Deepfakes, with XcepTemporal (CE) getting above 83% on these datasets. On the FaceSwap dataset, however, XceptionNet was significantly better – this marks the main exception to our overall findings that showed our models to be superior.

An interesting observation is that the test accuracies for the MPEG compression rise after retraining on datasets augmented just with the JPEG compression samples and not with the MPEG samples.

5.2.6 Transfer learning / Domain Adaptation.

As new deepfake generation algorithms are released, companion deepfake detection algorithms follow. In a phenomenon referred to as catastrophic forgetting, when new data samples are introduced into a training regiment, the performance on original samples suffers.

Dataset / Model	FaceForensics++ [62]						Celeb-DF [44]		
	Real	F2F	NT	DF	FS	Overall	Real	Fake	Overall
XceptionNet [12]	89.90	94.26	98.77	99.56	99.13	96.32	70.88	95.31	86.17
XceptionNet (KL)	**100**	**100**	**100**	**100**	**100**	**100**	93.21	98.25	96.89
XcepTemporal (CE)	98.95	98.57	**100**	**100**	**100**	99.83	92.77	99.81	96.78
XcepTemporal (KL)	**100**	**100**	**100**	**100**	**100**	**100**	**93.82**	99.02	97.18
XcepTemporal (EN)	83.22	97.75	95.31	**100**	**100**	95.25	93.12	**100**	97.51
XcepTemporal (EN_{1+n})	89.67	96.08	94.53	99.90	99.61	95.95	93.19	**100**	**98.01**

Table 7: **Transfer learning/Domain Adaptation:** Models are initialized on FaceForensics++ [62] and trained on Celeb-DF [44] along with 50 out of 720 Faceforensics++ videos of each class to prevent forgetting.

We train our model on FaceForensics++ and transfer learn to Celeb-DF. During transfer learning, all the layers but the classification layer (for CE variant) are frozen. After convergence, we lower the learning rate and fine-tune all layers. To prevent catastrophic forgetting, we include 50 (out of 720) randomly chosen samples from each FaceForensics++ deepfake type along with the Celeb-DF training data. The results are shown in Table 7. We note that after the final stage of training, our models perform well on all datasets, showing their ability to learn new types of fakes without forgetting how to classify original fakes.

5.2.7 Ablation Study

To help understand our model further, we explore several aspects and elucidate on the decisions involved in defining the model.

Recurrent layers. To understand the importance of recurrent layers in the XcepTemporal model, Table 8 shows the combinations between uni/bidirectional and single/double LSTM layers. The results in Table 8 suggest that the backward pass within a recurrent layer extracts meaningful temporal information. In addition, the added parameters from the secondary layer help improve results.

Recurrent Layers	FF++	Celeb-DF
Uni + Single	92.37	84.48
Uni + Double	92.9	86.02
Bi + Single	96.99	93.21
Bi + Double	**99.71**	**97.01**

Table 8: **Ablation:** Accuracy of XcepTemporal (CE) with different recurrent structures.

Length and Stride. We now examine the variation in performance for different lengths of the input sequence and different lengths of strides between two consecutive input sequences. Longer sequence lengths enable temporal features of longer duration but require more compute resources.

The combined accuracy on both the test sets shown in Table 1 for different lengths and

Length	Stride	Accuracy
4	4	98.02
8	8	99.24
8	6	98.81
16	16	99.29
16	12	99.16
32	32	**99.50**
32	24	99.37

Table 9: **Ablation:** Accuracy of XcepTemporal (CE) on the combined test sets for various length and stride combinations.

strides is shown in Table 9. While the overall accuracy does not change much, we chose a length and stride of eight because

1) shorter sample lengths translate into more training samples

2) shorter sample lengths are more capable at detecting short sequences of manipulated frames embedded between real frames.

5.3 RGB+EF XcepTemporal Model

5.3.1 Within-Domain Results

Unlike the CNN+Temporal model, RGB+EF XcepTemporal model requires preprocessed edge flow maps from the whole dataset for training and evaluation. To validate the performance of the model, DFDC-mini dataset was chosen from which the edge flow maps were extracted prior to training. As the isolated dataset contains only a couple of thousands of videos, a random 60-40 split for training and testing was carried out, while maintaining the ratio between the number of real and fake videos on both the sets. The variants of the model are trained alongside the best performing CNN+Temporal models and evaluated on the held out test set.

Model	Real	Fake	Overall
XcepTemporal (CE)	90.45	96.64	94.25
XcepTemporal (KL)	90.36	97.13	94.67
EffTemporal-B4 (CE)	90.23	97.57	94.65
EffTemporal-B4 (KL)	90.51	98.10	95.15
RGB+EF XcepTemporal (Early Fusion)	92.68	99.32	96.76
RGB+EF XcepTemporal (Mid-Fusion)	**95.37**	**99.53**	**97.94**
RGB+EF XcepTemporal (Late Fusion)	91.33	99.45	96.38

Table 10: **Within-Domain Results:** Our models, CNN+Temporal and RGB+EF XcepTemporal, and their variants are trained on the train set and evaluated on the test set of DFDC-mini dataset.

From Table 10, it can be observed that leveraging the edge and flow information from the video facilitates in better discrimination between the real and forged videos. Compared to FaceForensics++ [62] and Celeb-DF [44], DFDC [17] has a diversified range of videos where the subjects were exposed from low lighting to outdoor conditions, rapid camera movements and an exhaustive list of such factors. These aspects explain the relative drop in test accuracies across datasets.

5.3.2 Cross-Domain Results

A basic requirement in building a deepfake detector that can be employed in real-world application is to test its performance on unforeseen forgery. Although the RGB+EF XcepTemporal model accurately detects forged videos from the test set of the DFDC-mini dataset, we next test the trustworthiness of the model by testing on datasets not seen during training.

Model	Real	Fake	Overall
XcepTemporal (CE)	64.10	77.59	72.46
XcepTemporal (KL)	63.68	78.41	72.74
EffTemporal-B4 (CE)	67.64	78.89	74.38
EffTemporal-B4 (KL)	67.20	80.15	75.01
RGB+EF XcepTemporal (Early Fusion)	71.92	84.31	79.46
RGB+EF XcepTemporal (Mid-Fusion)	**73.33**	**86.81**	**81.29**
RGB+EF XcepTemporal (Late Fusion)	72.44	86.68	80.56

Table 11: **Cross-Domain Results:** Our models, CNN+Temporal and RGB+EF XcepTemporal, and their variants are trained on the FaceForensics++ [62] train set and evaluated on the DFDC-mini dataset.

We trained the model on FaceForensics++ dataset and tested it on the whole DFDC-mini dataset. From the results in Table 11, it is evident that incorporating the edgeflow input into the model significantly improves the forgery detection accuracy across domains. Although the deepfake generation process was different across the datasets, these edgeflow maps provide the model with rich beyond-the-visual instance-level flow attributes between successive frames that conceivably came into effect during the training. It can also be learned that the mid-fusion variant of the RGB+EF XcepTemporal model performed marginally better than its counterparts because its design provided the model with more favorable dynamics for semantic feature abstraction.

5.3.3 Immunizability on Adversarial Attacks

Our models use first-order visual features as inputs. To make our models resilient against any first-order attacks, the models must be aware of them in the first place. Some of the known perturbations to the images are blurring, compressing, cropping, luminating/darkening, and additive noise; or a random combination of them. As our models are trained on the Dlib [32] extracted and extended faces irrespective of the location of the face in the video, cropping doesn't significantly impact the performance of both of our models. Prior work has shown that manipulating the lighting in the video doesn't affect deepfake detectors in general because the datasets are diverse in the lighting spectrum. So, we choose to experiment with blur, compression, and noise on our models. For analysis purpose, we choose Gaussian noise and Gaussian blur out of numerous types that can be applied on the images.

For all the experiments, we arbitrarily administer one of the following options on each sample (as shown in Figure 9):

- Additive Gaussian Noise (random normal distribution) - $\mu = 0$ and $\sigma \epsilon [1, 5]$.
- Gaussian Blur with one of the kernel sizes - 3, 5, 7, or 9.

- JPEG Image Compression with a quality factor chosen from a range of 25 to 75.

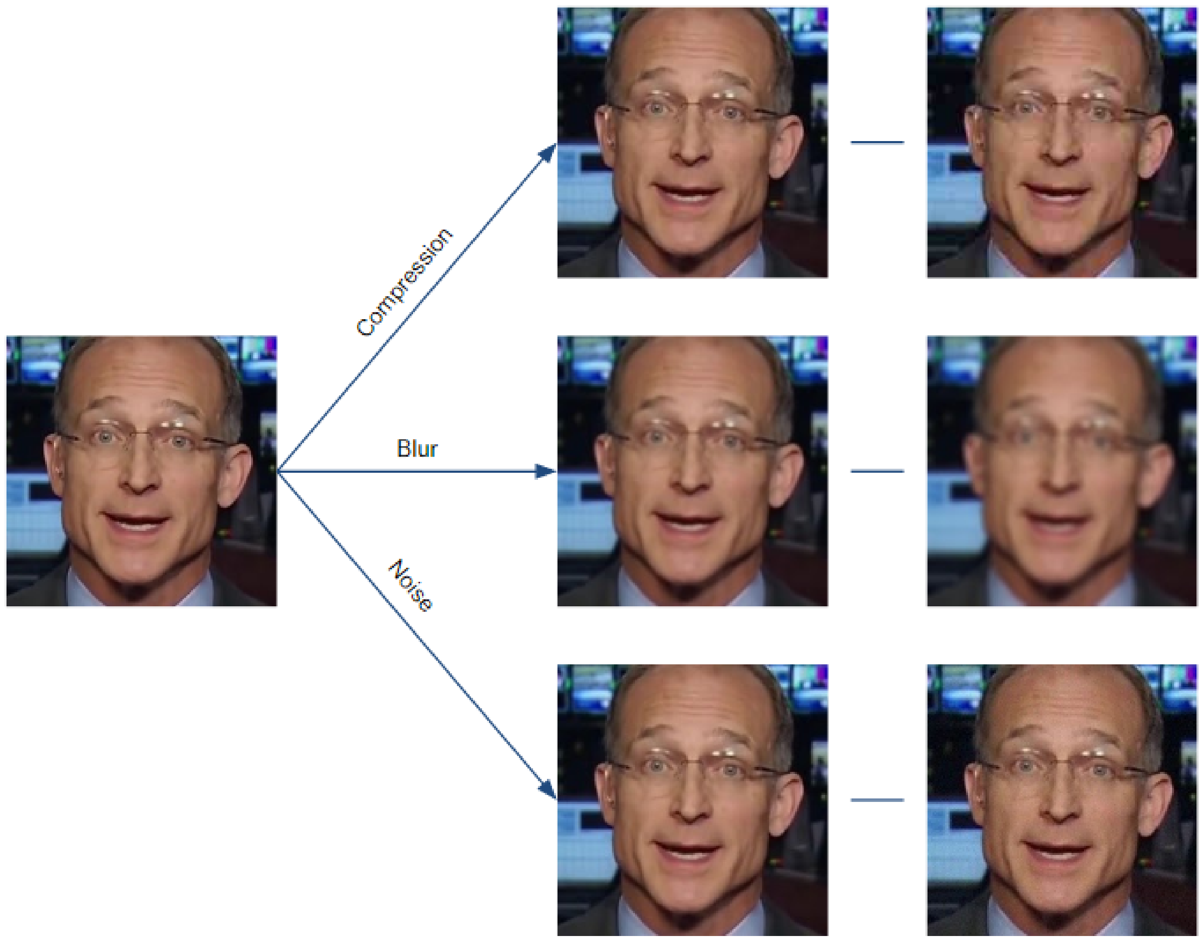

Figure 9: Visualization of the attacks on a test sample. The sample is administered with the least and most perturbations of the corresponding attacks exercised in the experiments.

For the combined column in Tables 12 and 13, we apply all the three attacks on all the samples during testing.

Before immunizing both our models against such attacks, they were tested on individual and combined attacks as shown in Table 12. The RGB+EF XcepTemporal variants show

Model	Blur	Compression	Noise	Combined
XcepTemporal (CE)	47.22	35.81	19.48	12.24
XcepTemporal (KL)	43.78	38.16	15.57	11.96
EffTemporal-B4 (CE)	48.12	53.68	18.21	10.54
EffTemporal-B4 (KL)	40.03	51.24	17.40	12.89
RGB+EF XcepTemporal (Early Fusion)	63.44	64.83	56.42	37.58
RGB+EF XcepTemporal (Mid-Fusion)	**67.18**	68.39	**70.30**	**45.75**
RGB+EF XcepTemporal (Late Fusion)	60.11	**69.10**	63.61	41.40

Table 12: **Pre-Immunization Results:** Our models, CNN+Temporal and RGB+EF XcepTemporal, and their variants are evaluated on the adversarial attacks administered on the whole DFDC-mini dataset. The accuracies reported are overall accuracy across the test set of the dataset.

Model	Blur	Compression	Noise	Combined
XcepTemporal (CE)	78.54	79.89	75.26	70.44
XcepTemporal (KL)	80.27	79.43	77.61	70.63
EffTemporal-B4 (CE)	82.16	81.74	78.04	73.73
EffTemporal-B4 (KL)	84.31	82.10	79.06	73.91
RGB+EF XcepTemporal (Early Fusion)	87.16	88.18	85.08	77.34
RGB+EF XcepTemporal (Mid-Fusion)	**88.32**	**91.69**	**89.31**	78.15
RGB+EF XcepTemporal (Late Fusion)	87.49	86.62	87.00	**78.41**

Table 13: **Post-Immunization Results:** Our models, CNN+Temporal and RGB+EF XcepTemporal, and their variants are evaluated on the adversarial attacks administered on the whole DFDC-mini dataset. The accuracies reported are overall accuracy across the test set of the dataset.

significant improvement in detection over the CNN+Temporal variants, specifically when the attacks are combined during inference.

The models are retrained on the augmented train set and evaluated on the test set of the

DFDC-mini dataset. The results are shown in Table 13. When immunized to the adversarial attacks, all of our models seemed to acclimatize comfortably. This experiment demonstrates the adaptability of our models to known attacks and that these models can be attuned to any future attacks when their mode of fabrication is ascertained.

6 Conclusion

Inspired by recent progress in digital forgery, we introduce two models, CNN+Temporal, a convolutional recurrent neural network framework for deepfake detection and RGB+EF XcepTemporal, an improvement over the XcepTemporal model where semantically rich features from the frame's edge and denseflow information are exploited in addition to the frames themselves. We use ImageNet [15] pretrained XceptionNet and EfficientNet features as salient and efficient facial feature representations. These representations are passed into bidirectional recurrent layers to aid in detecting temporal inconsistencies across the frames. Our model is trained both with traditional cross entropy and KL divergence loss functions. We demonstrate the robustness of our methods on the popular FaceForensics++ and Celeb-DF datasets, as well as the custom-created DFDC-mini dataset. Our methods obtain new benchmark standards and are shown to generalize well to unforeseen attacks. We reckon our work will be used as a baseline network in building more resilient models for detecting deepfakes and other analogous applications.

6.1 Future Work

This work presents a convolutional recurrent network to identify forged videos apart from unaltered videos. Taking this network as baseline, we employ the dense flow maps along with edge maps as additional input channels to provide the recurrent network with semantically richer features. However, as described in earlier sections, the model is sensitive to the visual aspects like blur and compression. As the proposed models only utilize visual input and its flow, there is plenty of room for improving on it. Some of the possible extensions to our proposed work are:

- Experimenting with mid-fusion at other stages of the XceptionNet architecture; fusing after all the eight separable convolutional blocks, for instance.

- Replacing XceptionNet with EfficientNet-B4 and defining multiple variants in the RGB+EF XcepTemporal model.

- Using the EdgeFlow (EF) as a segregated three-channel input (first map consists of edges, second map with the Δx of the dense optical flow, and the third map with the Δy of the dense flow) instead of a unified three-channel input.

- Syncing between the audio and the visuals because the audio in a forged video might not be consistent with the visuals.

- Using action units derived from the facial landmark movement between the frames to distinguish natural facial motion with that of computer generated.

- Behavior analysis using dense visual input and/or sparse facial landmarks.

- Manually extracting multiple degrees of motion and analyzing its potential as a feature set.

References

[1] Darius Afchar, Vincent Nozick, Junichi Yamagishi, and Isao Echizen. Mesonet: A compact facial video forgery detection network. In *WIFS*, 2018.

[2] Shruti Agarwal, Tarek El-Gaaly, Hany Farid, and Ser-Nam Lim. Detecting deep-fake videos from appearance and behavior. *arXiv preprint arXiv:2004.14491*, 2020.

[3] Shruti Agarwal, Hany Farid, Ohad Fried, and Maneesh Agrawala. Detecting deep-fake videos from phoneme-viseme mismatches. In *Proc. Conference on Computer Vision and Pattern Recognition Workshops*, 2020.

[4] Shruti Agarwal, Hany Farid, Yuming Gu, Mingming He, Koki Nagano, and Hao Li. Protecting world leaders against deep fakes. In *CVPR Workshops*, 2019.

[5] Grégoire Allaire, François Jouve, and Anca-Maria Toader. Structural optimization using sensitivity analysis and a level-set method. *Journal of computational physics*, 194(1):363–393, 2004.

[6] Burak Bekci, Zahid Akhtar, and Hazım Kemal Ekenel. Cross-dataset face manipulation detection.

[7] Nicolò Bonettini, Edoardo Daniele Cannas, Sara Mandelli, Luca Bondi, Paolo Bestagini, and Stefano Tubaro. Video face manipulation detection through ensemble of cnns. *arXiv preprint arXiv:2004.07676*, 2020.

[8] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

[9] Nicholas Carlini and Hany Farid. Evading deepfake-image detectors with white-and black-box attacks. *arXiv preprint arXiv:2004.00622*, 2020.

[10] Zehao Chen and Hua Yang. Manipulated face detector: Joint spatial and frequency domain attention network. *arXiv preprint arXiv:2005.02958*, 2020.

[11] Yunjey Choi, Minje Choi, Munyoung Kim, Jung-Woo Ha, Sunghun Kim, and Jaegul Choo. Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8789–8797, 2018.

[12] François Chollet. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017.

[13] Umur Aybars Ciftci and Ilke Demir. Fakecatcher: Detection of synthetic portrait videos using biological signals. *arXiv preprint arXiv:1901.02212*, 2019.

[14] Davide Cozzolino, Justus Thies, Andreas Rössler, Christian Riess, Matthias Nießner, and Luisa Verdoliva. Forensictransfer: Weakly-supervised domain adaptation for forgery detection. *arXiv preprint arXiv:1812.02510*, 2018.

[15] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.

[16] Joshua Denholtz. Deep fake detection: Non-facial feature cropping, and a novel conditional loss function.

[17] Brian Dolhansky, Russ Howes, Ben Pflaum, Nicole Baram, and Cristian Canton Ferrer. The deepfake detection challenge (dfdc) preview dataset. *arXiv preprint arXiv:1910.08854*, 2019.

[18] Ricaard Durall Lopez, Margret Keuper, Franz-Josef Pfreundt, and Janis Keuper. Unmasking deepfakes with simple features.

[19] Gunnar Farnebäck. Two-frame motion estimation based on polynomial expansion. In *Scandinavian conference on Image analysis*, pages 363–370. Springer, 2003.

[20] John Fletcher. Deepfakes, artificial intelligence, and some kind of dystopia: The new faces of online post-fact performance. *Theatre Journal*, 2018.

[21] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 2672–2680, 2014.

[22] David Güera and Edward J Delp. Deepfake video detection using recurrent neural networks. In *AVSS*, 2018.

[23] Sungjoo Ha, Martin Kersner, Beomsu Kim, Seokjun Seo, and Dongyoung Kim. Marionette: Few-shot face reenactment preserving identity of unseen targets. *arXiv preprint arXiv:1911.08139*, 2019.

[24] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.

[25] Geoffrey E Hinton, Alex Krizhevsky, and Sida D Wang. Transforming auto-encoders. In *ICANN*. Springer, 2011.

[26] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 1997.

[27] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer*

vision and pattern recognition, pages 4700–4708, 2017.

[28] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*, 2017.

[29] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4401–4410, 2019.

[30] Hyeongwoo Kim, Pablo Carrido, Ayush Tewari, Weipeng Xu, Justus Thies, Matthias Niessner, Patrick Pérez, Christian Richardt, Michael Zollhöfer, and Christian Theobalt. Deep video portraits. *ACM Transactions on Graphics (TOG)*, 2018.

[31] Taeksoo Kim, Moonsu Cha, Hyunsoo Kim, Jung Kwon Lee, and Jiwon Kim. Learning to discover cross-domain relations with generative adversarial networks. *arXiv preprint arXiv:1703.05192v2*, 2017.

[32] Davis E. King. Dlib-ml: A machine learning toolkit. *JMLR*, 10:1755–1758, 2009.

[33] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[34] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[35] Tomi Kinnunen, Md Sahidullah, Héctor Delgado, Massimiliano Todisco, Nicholas Evans, Junichi Yamagishi, and Kong Aik Lee. The asvspoof 2017 challenge: Assessing the limits of replay spoofing attack detection. 2017.

[36] Pavel Korshunov and Sébastien Marcel. Deepfakes: a new threat to face recognition? assessment and detection. *arXiv preprint arXiv:1812.08685*, 2018.

[37] Pavel Korshunov and Sébastien Marcel. A cross-database study of voice presentation attack detection. In *Handbook of Biometric Anti-Spoofing*, pages 363–389. Springer, 2019.

[38] Iryna Korshunova, Wenzhe Shi, Joni Dambre, and Lucas Theis. Fast face-swap using convolutional neural networks. In *ICCV*, 2017.

[39] Marek Kowalski. faceswap. `https://github.com/MarekKowalski/FaceSwap`.

[40] Akash Kumar and Arnav Bhavsar. Detecting deepfakes with metric learning. *arXiv preprint arXiv:2003.08645*, 2020.

[41] Lingzhi Li, Jianmin Bao, Hao Yang, Dong Chen, and Fang Wen. Faceshifter: Towards high fidelity and occlusion aware face swapping. *arXiv preprint arXiv:1912.13457*, 2019.

[42] Lingzhi Li, Jianmin Bao, Ting Zhang, Hao Yang, Dong Chen, Fang Wen, and Baining Guo. Face x-ray for more general face forgery detection. *arXiv preprint arXiv:1912.13458*, 2019.

[43] Yuezun Li, Ming-Ching Chang, Hany Farid, and Siwei Lyu. In ictu oculi: Exposing ai generated fake face videos by detecting eye blinking. *arXiv preprint arXiv:1806.02877*, 2018.

[44] Yuezun Li, Xin Yang, Pu Sun, Honggang Qi, and Siwei Lyu. Celeb-df: A new dataset for deepfake forensics. *arXiv preprint arXiv:1909.12962*, 2019.

[45] Ming-Yu Liu, Xun Huang, Arun Mallya, Tero Karras, Timo Aila, Jaakko Lehtinen, and Jan Kautz. Few-shot unsupervised image-to-image translation. *arXiv preprint arXiv:1905.01723*, 2019.

[46] Jaime Lorenzo-Trueba, Junichi Yamagishi, Tomoki Toda, Daisuke Saito, Fernando Villavicencio, Tomi Kinnunen, and Zhenhua Ling. The voice conversion challenge 2018: Promoting development of parallel and nonparallel methods. *arXiv preprint*

arXiv:1804.04262, 2018.

[47] Shao-An Lu. faceswap-GAN. `https://github.com/shaoanlu/faceswap-GAN`.

[48] Yongyi Lu, Yu-Wing Tai, and Chi-Keung Tang. Attribute-guided face generation using conditional cyclegan. *arXiv preprint arXiv:1705.09966v2*, 2018.

[49] Bruce D Lucas, Takeo Kanade, et al. An iterative image registration technique with an application to stereo vision. 1981.

[50] Falko Matern, Christian Riess, and Marc Stamminger. Exploiting visual artifacts to expose deepfakes and face manipulations. In *WACVW*. IEEE, 2019.

[51] Yisroel Mirsky and Wenke Lee. The creation and detection of deepfakes: A survey. *arXiv preprint arXiv:2004.11138*, 2020.

[52] Daniel Mas Montserrat, Hanxiang Hao, SK Yarlagadda, Sriram Baireddy, Ruiting Shao, Emily Bartusiak, Justin Yang, David Guera, Fengqing Zhu, Edward J Delp, et al. Deepfakes detection with automatic face weighting. *arXiv preprint arXiv:2004.12027*, 2020.

[53] Paarth Neekhara, Shehzeen Hussain, Malhar Jere, Farinaz Koushanfar, and Julian

McAuley. Adversarial deepfakes: Evaluating vulnerability of deepfake detectors to adversarial examples. *arXiv preprint arXiv:2002.12749*, 2020.

[54] Huy H. Nguyen, Fuming Fang, Junichi Yamagishi, and Isao Echizen. Multi-task learning for detecting and segmenting manipulated facial images and videos, 2019.

[55] Huy H Nguyen, Junichi Yamagishi, and Isao Echizen. Capsule-forensics: Using capsule networks to detect forged images and videos. *arXiv preprint arXiv:1810.11215*, 2018.

[56] Yuval Nirkin, Yosi Keller, and Tal Hassner. Fsgan: Subject agnostic face swapping and reenactment. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 7184–7193, 2019.

[57] Aviv Ovadya and Jess Whittlestone. Reducing malicious use of synthetic media research: Considerations and potential release practices for machine learning. *arXiv preprint arXiv:1907.11274*, 2019.

[58] Guim Perarnau, Joost Van De Weijer, Bogdan Raducanu, and Jose M Álvarez. Invertible conditional gans for image editing. *arXiv preprint arXiv:1611.06355*, 2016.

[59] Yao Qin, Nicholas Frosst, Colin Raffel, Garrison Cottrell, and Geoffrey Hinton. Deflecting adversarial attacks. *arXiv preprint arXiv:2002.07405*, 2020.

[60] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

[61] Scott Reed, Zeynep Akata, Xinchen Yan, Lajanugen Logeswaran, Bernt Schiele, and Honglak Lee. Generative adversarial text to image synthesis. In *International Conference on Machine Learning*, pages 1060–1069, 2016.

[62] Andreas Rössler, Davide Cozzolino, Luisa Verdoliva, Christian Riess, Justus Thies, and Matthias Nießner. Faceforensics++: Learning to detect manipulated facial images. *arXiv preprint arXiv:1901.08971*, 2019.

[63] Nataniel Ruiz and Stan Sclaroff. Disrupting deepfakes: Adversarial attacks against conditional image translation networks and facial manipulation systems. *arXiv preprint arXiv:2003.01279*, 2020.

[64] Ekraam Sabir, Jiaxin Cheng, Ayush Jaiswal, Wael AbdAlmageed, Iacopo Masi, and Prem Natarajan. Recurrent convolutional strategies for face manipulation detection in videos. *Interfaces (GUI)*, 3:1, 2019.

[65] Sara Sabour, Nicholas Frosst, and Geoffrey E Hinton. Dynamic routing between cap-

sules. In *NeurIPS*, 2017.

[66] Conrad Sanderson and Brian C Lovell. Multi-region probabilistic histograms for robust and scalable identity inference. In *International conference on biometrics*, pages 199–208. Springer, 2009.

[67] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *CVPR*, 2015.

[68] Yujun Shen, Jinjin Gu, Xiaoou Tang, and Bolei Zhou. Interpreting the latent space of gans for semantic face editing. *arXiv preprint arXiv:1907.10786*, 2019.

[69] Aliaksandr Siarohin, Stéphane Lathuilière, Sergey Tulyakov, Elisa Ricci, and Nicu Sebe. First order motion model for image animation. In *Advances in Neural Information Processing Systems*, pages 7135–7145, 2019.

[70] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[71] Saniat Javid Sohrawardi, Akash Chintha, Bao Thai, Sovantharith Seng, Andrea Hickerson, Raymond Ptucha, and Matthew Wright. Poster: Towards robust open-world detection of deepfakes. In *Proceedings of the 2019 ACM SIGSAC Conference on Com-*

puter and Communications Security, pages 2613–2615. ACM, 2019.

[72] Supasorn Suwajanakorn, Steven M Seitz, and Ira Kemelmacher-Shlizerman. Synthesizing obama: Learning lip sync from audio. *ACM Transactions on Graphics (TOG)*, 2017.

[73] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015.

[74] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.

[75] Mingxing Tan, Bo Chen, Ruoming Pang, Vijay Vasudevan, Mark Sandler, Andrew Howard, and Quoc V Le. Mnasnet: Platform-aware neural architecture search for mobile. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2820–2828, 2019.

[76] Mingxing Tan and Quoc V Le. Efficientnet: Rethinking model scaling for convolutional

neural networks. *arXiv preprint arXiv:1905.11946*, 2019.

[77] Ayush Tewari, Ohad Fried, Justus Thies, Vincent Sitzmann, Stephen Lombardi, Kalyan Sunkavalli, Ricardo Martin-Brualla, Tomas Simon, Jason Saragih, Matthias Nießner, et al. State of the art on neural rendering. *arXiv preprint arXiv:2004.03805*, 2020.

[78] Justus Thies, Michael Zollhöfer, and Matthias Nießner. Deferred neural rendering: Image synthesis using neural textures. *arXiv preprint arXiv:1904.12356*, 2019.

[79] Justus Thies, Michael Zollhofer, Marc Stamminger, Christian Theobalt, and Matthias Nießner. Face2face: Real-time face capture and reenactment of rgb videos. In *CVPR*, 2016.

[80] Francis Tom, Mohit Jain, and Prasenjit Dey. End-to-end audio replay attack detection using deep convolutional networks with attention. In *Interspeech*, pages 681–685, 2018.

[81] Luisa Verdoliva. Media forensics and deepfakes: an overview. *arXiv preprint arXiv:2001.06564*, 2020.

[82] Ting-Chun Wang, Ming-Yu Liu, Andrew Tao, Guilin Liu, Jan Kautz, and Bryan Catanzaro. Few-shot video-to-video synthesis. *arXiv preprint arXiv:1910.12713*, 2019.

[83] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Guilin Liu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. Video-to-video synthesis. arxiv 2018. *arXiv preprint arXiv:1808.06601*, 2018.

[84] Qiuxia Wu, Zhiyong Wang, Feiqi Deng, and David Dagan Feng. Realistic human action recognition with audio context. In *2010 International Conference on Digital Image Computing: Techniques and Applications*, pages 288–293. IEEE, 2010.

[85] Xi Wu, Zhen Xie, YuTao Gao, and Yu Xiao. Sstnet: Detecting manipulated faces through spatial, steganalysis and temporal features. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2952–2956. IEEE, 2020.

[86] Xin Yang, Yuezun Li, and Siwei Lyu. Exposing deep fakes using inconsistent head poses. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8261–8265. IEEE, 2019.

[87] Ning Yu, Larry S Davis, and Mario Fritz. Attributing fake images to gans: Learning and analyzing gan fingerprints. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 7556–7566, 2019.

[88] Egor Zakharov, Aliaksandra Shysheya, Egor Burkov, and Victor Lempitsky. Few-shot adversarial learning of realistic neural talking head models. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 9459–9468, 2019.

[89] Yang Zhou, DIngzeyu Li, Xintong Han, Evangelos Kalogerakis, Eli Shechtman, and Jose Echevarria. Makeittalk: Speaker-aware talking head animation. *arXiv preprint arXiv:2004.12992*, 2020.

[90] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. pages 2223–2232, 2017.

[91] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. *arXiv preprint arXiv:1703.10593v6*, 2017.