Deepfake Detection using Capsule Networks with Long Short-Term Memory Networks

Akul Mehra
Master Thesis
University of Twente, M-CS-DST
a.mehra@student.utwente.nl

***Abstract*—With the recent advancement of technology, particularly with graphics processing and artificial intelligence algorithms, fake media generation has become easier. Using deep learning techniques like Deepfakes and FaceSwap, anyone can generate fake videos by manipulating the face/voice of the target in the video. These AI synthesized face-swapping videos, also known as deepfakes are a big threat to the authenticity and trustworthiness of online information. These deepfakes can be used for malicious purposes like phishing scams and fake news. Detecting face tampering in realistic forged videos generated using a recent technique called Deepfake has become of utmost importance. Traditional image forensics techniques have lower performance in detecting face tampering in videos due to the compression which degrades the features and makes it difficult to identify video tampering.**

This paper provides an overview of the recent developments in deepfake technologies, how they are generated, what are the data-sets available. We discuss what inconsistencies are introduced in videos due to deepfake generation and propose a spatio-temporal hybrid model of Capsule Networks integrated with Long Short-Term Memory Networks. This model exploits the inconsistencies and identifies real and fake videos and is our contribution towards deepfake detection. Therefore, our research question for this paper is "Can the performance of detecting inconsistencies in a video to identify deepfakes be improved by combining a long short-term memory network with a capsule network to create a spatio-temporal hybrid model?". Firstly, a Capsule Network is introduced to detect spatial inconsistencies in a single frame and then combined with LSTM to detect the spatio-temporal inconsistencies across multiple frames and achieves an accuracy of 83.42%. The state-of-the-art model XceptionNet for deepfake detection is used as the baseline for a single frame detection and combined with LSTM for multiple frames detection. Using visualization of the capsule's activation, we understand what features the capsules learn and provide an explanation for identifying deepfakes and real videos. Using 3 different frame selection techniques, we also show that frame selection has a significant impact on the performance of our models. The model is tested on 2 additional data-sets with multiple deepfake generation methods, different augmentations, and different facial filters like dog filter and flower crown filter. Finally, we conclude the paper with an improvement in the Capsule Network's performance when combined with an LSTM but fails to outperform the state-of-the-art model on one data-set by a small gap of ~3%, whereas, achieves similar performance on another data-set having never-seen-before deepfake generation method and heavy augmentations. With almost comparable performance with the state-of-the-art model, in contrast to the size, our model has $1/5^{th}$ the number of parameters and $1/4^{th}$ the size of the state-of-the-art model and hence, is a lighter model and has reduced computational cost.

I. Introduction

Deepfake technology has found many applications like de-aging people, lip-sync, and face-swapping. These technologies are beneficial in the media industry, such as lip-sync can be used for dubbing a movie into another language, while keeping it realistic and entertaining for the viewers. Various deepfakes available on the web are usually where one movie actor's face like Nicolas Cage has been swapped onto the target actor's face in random movies [1]. Another recent example by a deepfake artist to make the actor Robert De Niro younger in the movie "The Irishman" where Netflix spent $175 million to de-age him, the deepfake artist made more superior de-aging results to Netflix's CGI within 7 days by using free deepfake technology [2]. Figure 1 shows an example of deepfake video generated by Facebook on how to make pour-over coffee [3].

Fig. 1: Deepfake video on how to make Pour-over Coffee.

Although the advancement in deep learning and deepfake technology has many beneficial applications in daily life, business, and film industry, they can also serve for malicious purposes and make people struggle to believe what is real. For example, anyone with deepfake technology can make a powerful politician an artificial intelligence (AI) puppet and make them say things, which they may have never said in real life. A recent example of this lip-sync technology was [4], where Barack Obama, the former president of USA, was seen saying bad things about the current president Donald Trump, was an impersonation done by the famous actor/comedian Jordan Peele. This video was made to show how realistic the deepfake technology has become and also, how it can influence future elections when used with wrong motives. A recent example of this was witnessed in action in Indian election

campaigns where the Delhi president for the Bharatiya Janata Party (BJP) was seen criticizing the incumbent Delhi government of Arvind Kejriwal [5]. The deepfake video went viral on Whatsapp, a cross-platform messaging app, and reached approximately 15 million people, dissuading them from voting for the rival political party in Delhi.

Deepfakes have already been used for fraudulent use-cases. [6] describes the first case where deepfake audio was used to scam a CEO of a UK-based energy firm and robbed €220,000. Another mistrust due to the rise of deepfakes is making people believe something real, is said to be fake, like a politician's video criticizing the rival party being real, but that politician refuses all the allegations and calls them fake videos. [7] is one example of such a case where simple video manipulations of the original video of Nancy Pelosi, the speaker of the US House of Representatives, made people believe it is a fake video generated using the deepfake technology. While it was a real video, the video's speed was slowed down by 25% and the pitch was altered to make it sound real, made her appear like she was slurring words, and was drunk. Due to the rise of deepfakes, it has been much easier to cause misinterpretation of videos, spread lies, and misinformation. Due to the easy access to the Internet, anyone can viral a deepfake video and before the video has been distinguished as real or fake, the damage is already done, making people believe what they want to, without knowing the truth. This kind of fake news, is making people lose trust in what is real and what is fake anymore, and how we can distinguish between the two. Hence, there is a requirement for research in this area which can help in detecting deepfakes from real and stopping them before they can spread misinformation.

This research aims to create a detector that can be used to identify a video as a deepfake or not. We perform analysis of the recent developments in deepfake technologies and propose a new methodology by integrating Capsule Networks (CapsuleNet) with Long Short-Term Memory (LSTM) Networks and our contribution towards deepfake detection. Therefore, the following research question is addressed in our work: **"Can the performance of detecting inconsistencies in a video to identify deepfakes be improved by combining a long short-term memory network with a capsule network to create a spatio-temporal hybrid model?"**

The research question can be formulated in the following sub-questions:

RQ1: What are the current technologies available for deepfake generation and detection?
- a) What are the methods used to generate fake media?
- b) What are the available deepfake data-sets?
- c) What are the methods available to detect fake media, the state-of-the-art model, and their benchmarks?

RQ2: How does a Capsule Network perform for detecting spatial inconsistencies using a single frame?
- a) Does training a Capsule Network using a single frame to detect spatial inconsistencies in a frame helps in identifying real and deepfakes?
- b) By visualizing the activation neurons/capsules, what inconsistencies are detected by Capsule Network to identify a sample as fake/real?
- c) How does the Capsule Network perform in comparison to the state-of-the-art Convolutional Neural Network (CNN)?

RQ3: Can the performance be improved by integrating the Capsule Network with a Long Short-Term Memory Network to also detect temporal inconsistencies?
- a) Does training a combined Capsule Network+LSTM network using a sequence of frames to detect spatial + temporal inconsistencies in a video helps in identifying real and deepfakes?
- b) Does the selection of frame sequences have a significant impact on the detection of fake videos?
- c) How does the Capsule Network+LSTM network perform in comparison to the state-of-the-art CNN+LSTM network?

In this paper, we first present the background on this topic, available data-sets, and techniques for deepfake detection in Section II. The approach of this research is defined in Section III, where we explain the data-set being used, the preprocessing pipeline, and our proposed model. We also describe the performance metrics and evaluation methods that will be used to evaluate our models and perform comparisons. The results of this research are shown and explained in Section IV, where we visualize the activations of capsules to explain what spatial features are learned by the capsule network, and performance comparisons of our model to baseline benchmarks. Finally, in Section V, we conclude with our findings, and the research questions are answered.

II. Background

A. Fake Media Generation

Face Swap: This manipulation means swapping a face from a source video onto the face on the target video. Deepfake is the face swaps performed by using deep learning. FakeApp [8] is the first attempt towards generating fake videos. The model behind FakeApp is an autoencoder-decoder pairing structure, where the autoencoder extracts the latent features of the face, and the decoder reconstructs the face from these features. Two encoder-decoder pairs are used to train on the source set and the target set respectively. The encoders share the weights such that the two pairs share the same encoder. When the shared encoder passes the features of image A to the decoder of image B, it performs the face swap. This approach is used in various deepfake generation works such as FaceSwap [9], [10], DeepFaceLab [11], DFaker [12].

Facial Expression: Manipulating facial expressions of a person, i.e. transferring the facial expression from one person to another person. Face2Face [13], a real-time facial reenactment system is one of the popular techniques where the facial expressions of a source face are transferred onto the target's face. One of the recent examples is where a realistic video is

generated of a person (Barack Obama) [4], [14], and shown speaking things he never said.

Facial Attributes: These manipulations include modifying facial attributes such as gender, the color of skin or hair, age, adding glasses, etc. One of the recent examples is using DeepFaceLab [11], generated for performing face swap, which is used for de-aging an actor (Robert De Niro) in the movie "The Irishman" [15].

B. Deepfake Data Set

UADFV [16] is one of the first deepfake data-set and contains 49 real videos collected from YouTube and 49 deepfake videos. The videos are synthesized using the DNN model using FakeApp.

DF-TIMIT [17] is a data-set generated using Faceswap-GAN applied to the Vid-TIMIT data-set. It consists of 640 deepfake videos split into two equal subsets: DF-TIMIT-LQ and DF-TIMIT-HQ, with the size of synthesized faces of 64 x 64 and 128 × 128 pixels, respectively.

FaceForensics++ [18] contains 1,000 real videos collected from YouTube and 1,000 deepfake videos generated using Faceswap.

Deepfake Detection [19] Data-set provided by Google/Jigsaw consists of 363 real videos and 3068 deepfake videos of 28 consented individuals of various genders, ages, and ethnic groups. The deepfake synthesis method is not disclosed.

Celeb-DF [20] provides a new data-set generated using an improved deepfake synthesis method, using videos of 59 celebrities collected from YouTube to make 5639 deepfake videos and 590 real videos.

Deepfake Detection Challenge Preview [21] is a preview data-set provided by the partnership of various organizations including Facebook, Microsoft, AWS, and Partnership on AI. The data-set is constructed via a data collection campaign where 66 consented individuals from various genders, ages, and ethnic groups are used to make a more diverse and real-world deepfakes. The preview data-set consists of 4119 deepfake videos, generated using two different unknown synthesis algorithms and 1131 real videos.

Deepfake Detection Challenge [22], [23] is a data-set consisting of 100,000 fake videos and 19,154 real videos released on Kaggle [24], an online platform where data science and machine learning practitioners compete. The challenge was released with prize money of $1,000,000 to contribute and develop better deepfake detection models. The deepfake videos are generated using 4 different techniques, namely Deepfake Autoencoder (DFAE), MM/NN face swap, Neural Talking Heads (NTH), and Face Swapping GAN (FSGAN).

A concise summary of the available deepfake data-sets is provided in Table I.

Database	Real Videos	Fake Videos
UADFV (2018) [16]	49 (YouTube)	49 (FakeApp)
DeepfakeTIMIT (2018) [17]	320 (VidTIMIT)	640 (faceswap-GAN)
FaceForensics++ (2019) [18]	1,000 (YouTube)	1,000 (DeepFake)
Deepfake Detection (2019) [19]	363 (Actors)	3,068 (DeepFake)
Celeb-DF (2019) [20]	590 (YouTube)	5,639 (DeepFake)
DFDC Preview (2019) [21]	1,131 (Actors)	4,119 (Unknown)
DFDC (2019) [22], [23]	19,154 (Actors)	100,000 (4 Different Methods)

TABLE I: Available Deepfake Data-sets

C. Fake Media Detection

The earlier generation of deepfake videos was not as realistic as the new deepfakes. These early deepfakes usually showed various kinds of physical inconsistencies and earlier detection models took advantage of these inconsistencies to detect fakes.

EyeBlinking [25] discusses inconsistencies like lack of blinking of eyes, were due to not enough training videos with the blinking of eyes. The average blinking rate of a normal human being is 10/min, the original videos have 34.1/min blinks whereas fake videos have only 3.4/min blinks. Long-term recurrent convolutional neural networks (LRCN) model is used to capture temporal dependencies and achieves the best performance of AUC=0.99 in comparison to CNN AUC=0.98 and eye aspect ratio EAR=0.79 on the UADFV data-set.

Two-Stream [26] is a two-stream network in which a CNN stream GoogLeNet InceptionV3 model [27] used for face manipulation detection is fused with a path triplet stream trained using steganalysis, the study of detecting features hidden using steganography, with SVM for classification and achieves AUC=0.927 on their data-set.

MesoNet [28] is another CNN based model that is designed with a lower number of layers to focus on the mesoscopic properties of images. As microscopic analysis based on image noise is not possible due to compression of videos and at the macroscopic level, human eyes struggle to distinguish forged images, hence they adopt an intermediate method to focus on mesoscopic. It provides two different models called Meso-4, which uses conventional convolutional layers and MesoInception-4 based on Inception modules [29]. The evaluation of these models is done on deepfake videos collected from the Internet and is achieved an average detection rate of 98% for Deepfake videos.

ConvolutionalLSTM [30] is a temporal-aware pipeline to identify deepfakes generated and validated on videos collected from multiple video-hosting websites. The proposed model consists of a combination of a CNN, for frame feature extraction combined with an LSTM for temporal sequence analysis. As the deepfakes are generated frame-by-frame, each frame has a new face generated which will have inconsistencies when compared to every other frame and therefore, lacks temporal awareness between frames. These temporal incon-

sistencies such as flickering in frames and inconsistent choice of illuminants are used to detect deepfakes and results in an accuracy of ~97% on their data-set.

HeadPose [16] is another example where inconsistent head poses are exploited. They compare two head poses estimated using only the central region landmarks and using the whole face landmarks. The difference between the two head poses is less for original videos as both should be consistent, whereas, significant for fake videos because the central region is generated and has different head pose orientation, while the outer region is the same. An SVM model based on an estimated 3D head pose orientation is used to identify as deepfakes and achieves AUC=0.890 on the UADFV data-set. These videos were easier to identify deepfake or not. With the improvement in techniques for synthesizing deepfakes, the newer data-sets are generally much better in both quality and quantity as compared to the first generation of deepfakes.

FaceWarpingArtifacts [31] uses VGG16 and ResNet based models [32] trained on face images manipulated by basic image processing functions like resizing and interpolating, to expose the face warping artifacts that usually happen due to the manipulations by deepfake generation. ResNet-50 achieves the best performance with AUC=0.974 on the UADFV data-set and outperforms HeadPose, Two-Stream, and MesoNet models.

XceptionNet classifier [33], is a traditional CNN with pre-trained weights of ImageNet. [18] provides an overview of the detection performance, where multiple models using steganalysis and CNN based networks are evaluated on the FF++ (FaceForensics++) data-set [18]. XceptionNet performs best in all face manipulation techniques and achieves the state-of-the-art accuracy of 96.36% for deepfakes. Followed by MesoNet [28], an InceptionNet [29] based CNN architecture to detect face tampering in videos with 87.27% accuracy. XceptionNet is also evaluated using the DFDC Preview data-set [21] and achieves 93.0% precision.

VisualArtifacts [34] focuses and exploits the missing reflections, details in the eyes, teeth, and facial contours of videos to detect deepfakes. It provides two models, one based on a multi-layer neural network model and another based on a logistic regression model to train and evaluate deepfake videos collected from YouTube. The multi-layer neural network achieves the best result with AUC=0.851 when both eyes and teeth are used for detecting deepfakes.

Multi-task [35] provides a CNN based model to simultaneously detect manipulated images and segment the manipulated areas as a multi-task learning problem and achieves the highest score on the segmentation task on the FF++ data-set.

RecurrentConvolutional [36] exploits the temporal discrepancies across multiple frames caused by the manipulations that are performed frame-by-frame. The CNN models used are DenseNet and ResNet, where DenseNet outperforms ResNet. The difference with ConvolutionalLSTM [30] is that they use pre-trained CNNs while RecurrentConvolutional models are trained end-to-end. DenseNet with alignment and bidirectional recurrent network achieves the best performance with accuracy=96.9% on the FF++ data-set.

Capsule [37] uses capsule structures [38] for deepfake detection. The architecture is based on a previous paper that used capsule networks for forgery detection and forensics [39]. The model uses the VGG19 [40] network as the backbone for deepfake detection. Although CapsuleNet achieves 92.17% accuracy and XceptionNet achieves 94.81% for deepfakes in multi-class classifications, CapsuleNet has a more balanced performance for all labels in FF++ data-set.

Although the above methods focused on generalized deepfake detection models, there are papers which focus on learning features of specific individuals. **ProtectingWorldLearders** [41] builds a one-class SVM based detection model to distinguish one individual from other individuals. Each individual's action units (AU) are extracted as features and learned to distinguish between an individual like Barack Obama vs others including deepfakes, comedy impersonators, etc.

A concise summary of the Deepfake detection methods and their best performance is provided in Table II.

D. Capsule Network

Although CNNs perform well in the domain of computer vision, they have limitations when applied to inverse graphics. The pooling layers in CNNs cause loss of information after every convolution layer and have local translational invariance due to which they are unable to identify the position of one object relative to another and predict a face if the important features are present but not in the correct position. Hinton et al. [42] in 2011 addressed these limitations and proposed the capsule architecture to overcome these drawbacks. Due to a lack of efficient algorithms and hardware limitations, the capsule network was not implemented effectively. With the recent developments of dynamic routing [38] and expectation-maximization routing [43] algorithms introduced in 2017 and 2018 respectively, capsule networks have been implemented with remarkable results and outperform CNNs on few object classification tasks. In [38], capsule network achieves 79% accuracy on affine test set whereas the traditional CNN model with a similar number of parameters achieves 66% accuracy. In [43], capsule network achieves significantly better accuracy and reduces the number of errors by 45% in comparison to the state-of-the-art CNN model. These developments introduced 1) dynamic routing-by-agreement and replaced the max-pooling of CNN, and 2) squashing which replaced the scalar output feature detectors of CNN with vector output capsules. The agreement between capsules which preserves the pose information enables the capsule networks to enclose more information than a CNN with less training data required.

Various applications of capsule networks have been seen in object classification and computer vision. Iesmantas et al. [44] perform a binary classification for breast cancer images using capsule networks. Saqur et al. [45] propose CapsGAN for generating images in the 3D domain using GANS. Duarte et al. [46] use capsule networks for video segmentation and in [47] perform action classification in videos. Capsule networks

Study	Features	Classifiers	Databases	Best Performance
Zhou et al. (2018) **Two-Stream** [26]	Image related Steganalysis	CNN, SVM	UADFV	AUC = 85.1%
			DeepfakeTIMIT (LQ)	AUC = 83.5%
			DeepfakeTIMIT (HQ)	AUC = 73.5%
			FF++ / DFD	AUC = 70.1%
			Celeb-DF	AUC = 53.8%
Afchar et al. (2018) **MesoNet** [28]	Mesoscopic Level	CNN	Own	Acc. = 98.4%
			UADFV	AUC = 84.3%
			DeepfakeTIMIT (LQ)	AUC = 87.8%
			DeepfakeTIMIT (HQ)	AUC = 68.4%
			FF++ (DeepFake, LQ)	Acc. $\simeq$ 90.0%
			FF++ (DeepFake, HQ)	Acc. $\simeq$ 94.0%
			FF++ (DeepFake, RAW)	Acc. $\simeq$ 98.0%
			Celeb-DF	AUC = 54.8%
Guera and Delp (2018) **ConvolutionalLSTM** [30]	Image + Temporal Informationl	CNN + RNN	Own	Acc. = 97.1%
Yang et al. (2019) **HeadPose** [16]	Head Pose Estimation	SVM	UADFV	AUC = 89.0%
			DeepfakeTIMIT (LQ)	AUC = 55.1%
			DeepfakeTIMIT (HQ)	AUC = 53.2%
			FF++ / DFD	AUC = 47.3%
			Celeb-DF	AUC = 54.6%
Li et al. (2019) **FaceWarpingArtifacts** [31]	Face Warping Artifacts	CNN	**UADFV**	**AUC = 97.4%**
			DeepfakeTIMIT (LQ)	**AUC = 99.9%**
			DeepfakeTIMIT (HQ)	**AUC = 93.2%**
			FF++ / DFD	AUC = 80.1%
			Celeb-DF	AUC = 56.9%
Rossler et al. (2019) **XceptionNet** [18]	Image-related Steganalysis	CNN	UADFV	AUC = 91.2%
			DeepfakeTIMIT (LQ)	AUC = 95.9%
			DeepfakeTIMIT (HQ)	AUC = 94.4%
			FF++ (DeepFake, LQ)	Acc. $\simeq$ 94.0%
			FF++ (DeepFake, HQ)	**Acc. $\simeq$ 98.0%**
			FF++ (DeepFake, RAW)	**Acc. $\simeq$ 100.0%**
			Celeb-DF	**AUC = 65.5%**
Matern et al. (2019) **VisualArtifacts** [34]	Visual Artifacts	Logistic Regression, MLP	Own	AUC. = 85.1%
			UADFV	AUC = 70.2%
			DeepfakeTIMIT (LQ)	AUC = 77.0%
			DeepfakeTIMIT (HQ)	AUC = 77.3%
			FF++ / DFD	AUC = 78.0%
			Celeb-DF	AUC = 55.1%
Nguyen et al. (2019) **Multi-task** [35]	Image-related	Autoencoder	UADFV	AUC = 65.8%
			DeepfakeTIMIT (LQ)	AUC = 62.2%
			DeepfakeTIMIT (HQ)	AUC = 55.3%
			FF++ / DFD	AUC = 76.3%
			Celeb-DF	AUC = 54.3%
Dolhansky et al. (2019) **XceptionNet** [21]	Image-related	CNN	**DFDC Preview**	**Precision = 93.0%** **Recall = 8.4%**
Sabir et al. (2019) **RecurrentConvolutional** [36]	Image + Temporal Information	CNN + RNN	**FF++ (DeepFake, LQ)**	**AUC = 96.9%**
Nguyen et al. (2019) **Capsule** [37]	Image-related	Capsule Network	UADFV	AUC = 61.3%
			DeepfakeTIMIT (LQ)	AUC = 78.4%
			DeepfakeTIMIT (HQ)	AUC = 74.4%
			FF++ (Deepfake)	Acc. = 92.17%
			Celeb-DF	AUC = 57.5%

TABLE II: Performance of Deepfake Detection Methods

are also used for forensics and forgery detection. Nguyen et al. [39] proposed capsule-forensics for detecting manipulated images and performance was better than MesoNet. Their next paper [37] proposes an improved capsule-forensics network for detecting fake images and videos generated on the FF++ dataset. The model achieved equivalent or better scores in comparison to state-of-the-art methods while using fewer parameters and hence, less computational cost. These advancements in capsule networks in multiple domains and forgery detection have motivated us to study and work with capsule networks for the detection of deepfakes and explain the theory behind it through visualizations.

E. Inconsistencies and Motivation

The idea behind using capsule networks is 1) they are more robust, preserve pose information, and are equivariant in parameters like translation, rotation, scale, thickness, etc. making them overcome the limitation of a CNN's not being rotational invariant and therefore, not requiring to feed rotated face images to the network. The capsule network learns the rotation as one of its parameters, making them not only detects feature but also their orientation. A rotated face has eyes, nose, and mouth features in the same orientation. 2) a Capsule Network requires fewer parameters than a CNN

while achieving similar performance. Spatial inconsistencies in deepfakes are usually blurred and flickering of faces, lack of sharpness in teeth region and no reflection in the eyes, also as the face is generated and moved on the target face, there are inconsistencies in boundary regions around the face generated which helps in identifying if a face is real or not. Therefore, training a deep learning model to detect these inconsistencies will help in identifying deepfakes. Hence, we'll be exploring a capsule network as the deep learning model in place of a traditional CNN for detecting spatial inconsistencies in videos to identify deepfakes as done in [37], [39].

As we are working with videos, and the deepfakes tamper with the face in the video by performing modifications frame-by-frame, every new face generated in a frame is different from other frames as the generation algorithm doesn't keep track of previous faces generated before generating a new face in the next frame. This creates temporal inconsistencies within frames across time, which can be exploited by training a recurrent neural network as they have the feature to remember the previous inputs as well, to detect these temporal inconsistencies and identify deepfakes as done in [30], [36].

Therefore, the main contribution of this work is by integrating a capsule network with an LSTM network. We propose a spatio-temporal hybrid model that will exploit and detect the inconsistencies in both the spatial domain and temporal domain and identify a video as a deepfake or not.

III. Method

In this section, we describe the data-set in detail, the preprocessing performed for extracting faces from frames, and our proposed method for detecting video face manipulations, i.e., given a sequence of frames as input, detect whether faces are real (pristine) or fake. We also describe the performance metrics used and our evaluation methods. The overall pipeline of our model is provided in Figure 2.

A. Data Set

The complete DFDC data-set provided on the Kaggle challenge is used [22], [24]. AWS, Facebook, Microsoft, and the Partnership on AI along with other academics have come together to build this Deepfake Detection Challenge to develop intelligent models that will help to detect real and manipulated media content. A preview data-set [21] was released initially with 66 paid actors, 1131 real and 4119 fake videos. Two different unknown facial modification approaches were used to generate fake videos. The complete DFDC data-set [23] contains over 470GB of videos (19,154 real videos and 100,000 fake videos) using 486 actors. Each video has a duration of ~10sec and is generated using 4 different deepfake generation techniques, namely Deepfake Autoencoder (DFAE), MM/NN face swap, Neural Talking Heads (NTH) and Face Swapping GAN (FSGAN), with no augmentations performed. The data-set is split into train, validation, and test set, which is used for training our model and evaluate performance. Additionally, [23] provides two test sets that are used for performance comparison on Kaggle, namely for the Public Leaderboard using a Public Test Set and Private Leaderboard using a Private Test Set. Public Test Set is collected in the same way as DFDC and contains 4,000 videos (2,000 real videos and 2,000 fake videos) from 214 actors who are not used in the DFDC data-set. The major difference from the DFDC data-set is one additional deepfake generation technique, StyleGAN is used along with the 4 generation techniques used for DFDC data-set, and heavy augmentations are applied to approximately 79% of all videos. Private Test contains 10,000 videos (5,000 real videos and 5,000 fake videos) and is focused on more generalization of videos and contains 50% videos collected similarly as the DFDC data-set using 260 actors along with 50% organic content collected from the Internet. Due to privacy issues and lack of consent from the actors in the 50% organic content collected from the Internet, these videos are not provided in the Private Test Set and therefore, are not taken into consideration. The final Private Test contains 5,000 videos (2,500 real videos and 2,500 fake videos). The data was generated using the same 4 techniques used for the DFDC data-set and heavy augmentations are applied to approximately 79% of all videos with additional never-before-seen filters including a dog filter and a flower crown filter.

In the rest of the paper, we reference the DFDC's test set as **DFDC Test Set**, Kaggle's Public test set as **Public Test Set**, and Kaggle's Private test set as **Private Test Set**.

Data-set	Total videos	Real videos	Fake videos	Augmentations
DFDC	119,154 (Actors)	19,154	100,000	No Aug
Public Test Set	4,000 (Actors)	2,000	2,000	79% Aug
Private Test Set	5,000 (Actors)	2,500	2,500	79% Aug + additional filters

TABLE III: Different Data-sets Used

For creating the data-set, we perform the following steps:

1) Subsampling: As the data-set is imbalanced with 100,000 fake videos and only 19,154 real videos, we randomly subsample our fake videos with random seed=84 such that the final data-set is balanced with 19,154 fake and 19,154 real videos.

2) Train-Test Split: As the DFDC data-set is provided in 50 parts, we perform folder wise split to avoid mixing of actors videos across multiple folders such that the model will be predicting scores on actors not seen before. We use folders 0-39 for Training, 40-44 for Validation, and 45-49 for Testing. Additionally, as multiple faces are possible in videos, and not all the faces in a video may be deepfakes, therefore to train the model with only deepfakes, we filter out the multiple face videos and only use single face videos for training.

After performing the above data-set creation steps, our data-set is split into the train (~70%), validation (~15%), and test-set (~15%). This data-set is then used to train our deepfake detection model and its performance is evaluated.

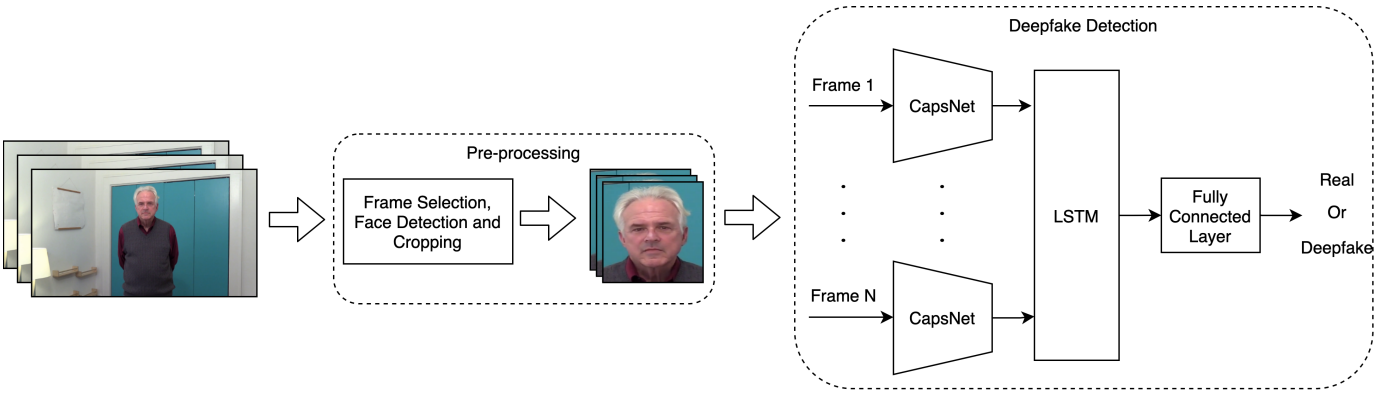


Fig. 2: Model Pipeline for Deepfake Detection.

B. Preprocessing

It is important to extract frames and perform pre-processing on the videos to reduce computational complexity. The pre-processing steps can be summarized as follows:

1) Frame Selection: Every video has 30 frames per second and ~10sec long i.e. ~300 frames, using all the frames is both computationally expensive and resource consuming. Therefore, we select 10 frames from each video to be used to train and evaluate the models. We perform three different methods of frame selection such that we can compare the performance of our model on each selection method and see whether the selection of frames impacts our results. Additionally, we select a single frame from each video to compare the impact of a single frame and multiple frames on our results. The 4 methods of selection are:

1) **Single Frame**: Extract the 10^{th} frame from each video to use towards a single frame spatial based model.
2) **First 10**: Extract the first 10 frames from each video as shown in Figure 3(a).
3) **Equal Interval**: Extract 10 frames from each video with a 1-sec interval as shown in Figure 3(b).
4) **Most Changes**: Select the 1-sec frame interval which has most changes happening within that interval in the following way:
   a) Extract 10 frames from each video with a 1-sec interval.
   b) Calculate the structural similarity (SSIM) between two consecutive frames for all the 10 frames.
   c) As SSIM provides a measurement for the similarity between two images, we select the consecutive frames pair which has the least structural similarity.
   d) Using the selected frame pair interval as start and endpoint respectively, extract 10 frames within them at an equal interval including the start and endpoint to get a series of 10 frames with most changes appearing in the face region in a 1-sec interval as shown in Figure 3(c).

2) Face detection and cropping: As we want to focus on the task of deepfake detection, we would like to only detect the face part with the facial manipulations.

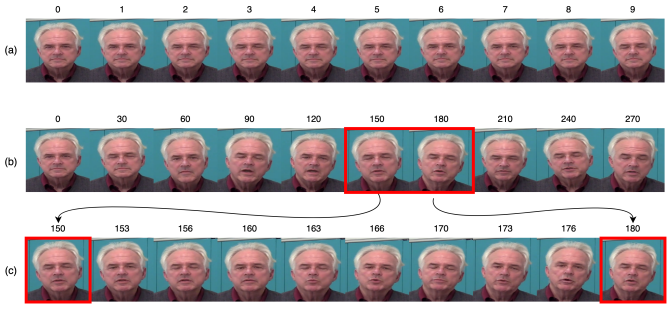


Fig. 3: Frame selection methods: (a) First 10: First 10 frames are selected (b) Equal Interval: 10 frames extracted at equal interval from ~10sec (~300 frames) video. (c) Most Changes: Frame 150 and Frame 180 have the least similarity score. 10 frames extracted between Frame 150 and Frame 180.

1) Face detection: We use a deep learning based face detection model called Mobilenet SSD [48] as it has higher performance with lower computation cost.
2) Selecting the same face across multiple frames: When detecting multiple faces across multiple frames, faces are swapped while detection due to different faces having higher confidence in different frames. To avoid this and keep all the faces uniform, we keep a bounding box for each face found in a frame and search for faces in that bounding box across all frames. If there is a miss in any faces across multiple frames, we drop that face from the video. This also avoids false detection of the face across multiple frames.
3) Crop face: As only facial features correspond to facial manipulations cropping the image only to focus on the face reduces the complexity of the model and should improve the model performance. Adding pixel padding to the crop window in a range of 1.5-2.0 times the crop window helps in capturing the spatial differences around the face boundaries. We use pixel padding=1.7 times the width of the face, such that the total width of the cropped

face becomes 1.7 times the actual cropped face.

4) Rescale: After cropping, as all the images should be of uniform size, we rescale them to the same scale. We use 224x224 as the image size.

3) Augmentation: As augmentation increases data and also makes the model robust and is also performed on the unseen Test data-set on Kaggle, we perform similar augmentations as performed in [21] i.e. Jpeg-Compression and downscale augmentation along with additional basic augmentations such as horizontal flip, rgbshift, brightness, contrast, gamma, hue and saturation provided by Albumentations [49]. We perform one of the following augmentations from Table IV on 33% of the training data:

Augmentation	Parameters	Probability
RandomBrightnessContrast	Brightness Limit = 0.3 Contrast Limit = 0.5	p = 0.1665
RandomGamma	Gamma Limit = (80, 120)	p = 0.1665
RGBShift	R Shift Limit = 105 G Shift Limit = 45 B Shift Limit = 40	p = 0.1665
HueSaturationValue	Hue Shift Limit = 42 Saturation Shift Limit = 10 Value Shift Limit = 17	p = 0.1665
JpegCompression	Quality Lower = 50 Quality Upper = 70	p = 0.334

TABLE IV: Augmentations Performed

Additionally, we perform Horizontal Flip with a 50% chance to flip and normalize images using the mean = (0.485, 0.456, 0.406) and standard deviation = (0.229, 0.224, 0.225).

C. Proposed Model

We propose our CapsuleNet + LSTM model which will detect spatio-temporal inconsistencies from a given sequence of frames. The model can be split into two part: the CapsuleNet, which acts as a feature extractor and identifies spatial inconsistencies in a single frame and the LSTM, which takes a sequence of feature vectors extracted by CapsuleNet as input from a sequence of frames and identifies temporal inconsistencies across the given sequence of frames.

For the Capsule Network, we use the capsule forensics model [39] and remove the output capsules to extract feature vectors as output. The model uses part of the pre-trained VGG-19 [40] (until the third max-pooling layer) as a feature extractor and is equivalent to the CNN part of the original capsule network architecture. After the features are extracted from the CNN, they are passed to multiple capsules, each with different weights initialized (initialized from a normal distribution). As using fewer capsules limits to capture fewer features [39], and using more capsules runs out of features to be learned as seen in our experiments and may overfit, we limit our model to use 10 capsules. Each capsule consists of a 2D convolutional part, a statistical pooling, and a 1D convolutional part. For the forensics task, the statistical pooling layer [50], [51] has been proven to be effective and also improves performance in our experiments. The statistical pooling layer includes mean and variance filters which are calculated as follows:

$$\text{Mean, } \mu_k = \frac{1}{H\text{x}W}\sum_{i=1}^{H}\sum_{j=1}^{W} I_{kij}$$

$$\text{Variance, } \sigma_k^2 = \frac{1}{H\text{x}W-1}\sum_{i=1}^{H}\sum_{j=1}^{W}(I_{kij}-\mu_k)^2$$

where H and W are the height and width of the filter respectively, k is the layer index, and I is the 2-dimensional filter array. The output of the statistical pooling layer is 1-dimensional, which is then passed through the 1D convolutional part and the final output is a feature vector of size 8 from a single capsule. Using 10 capsules and flattening, we obtain 80 features extracted from a single frame. These features help in detecting spatial inconsistencies in a given frame.

We perform the same with 10 frames of a video to get 10 feature vectors of size 80 each. These 10 feature vectors are then given as an input into a single layer LSTM model with 512 hidden units which captures the temporal inconsistencies across multiple frames using these feature vectors. The output of the last LSTM cell is then passed through a Fully Connected Layer of output size 256 followed by ReLU and Dropout of 50% to avoid overfitting in LSTM. The output is again passed through a second Fully Connected Layer and then through Softmax which provides a probability score between 0 (real) and 1 (fake). Using a more Fully Connected Layer helps in mapping complex features to output with better performance. As the model uses both spatial and temporal features to identify a given sequence of frames from a video as real or fake, hence, it is a spatio-temporal model for deepfake detection. The detailed model architecture is shown in Figure 4.

Cross-entropy loss function and AdamW optimizer [52] are used with a learning rate of 1e-3 and a weight decay of 1e-4 to optimize the network.

D. Baseline Models

1) CapsuleNet: The Capsule Forensics model [39] is used as the CapsuleNet. The number of capsules is set the same as our proposed model, i.e. 10 capsules. The input is a single frame and the output is the probability of the video being real or fake.

2) XceptionNet: XceptionNet model [18] is the state-of-the-art model for deepfake detection. We use the pre-trained model and replace the last layer with a set of custom layers. The custom layers are Fully Connected layer 2048 to 512, ReLU, BatchNorm1d, Dropout(0.5), Fully Connected layer 512 to 1 output. This output node is passed through Sigmoid to get a probability from 0 (real) to 1 (fake). Binary cross-entropy loss function and AdamW are used to optimize the network.

3) XceptionNet + LSTM: Additionally, we compare if combining XceptionNet with the LSTM model improves the performance of the state-of-the-art model. The pre-trained XceptionNet model is used with the last layer removed. The

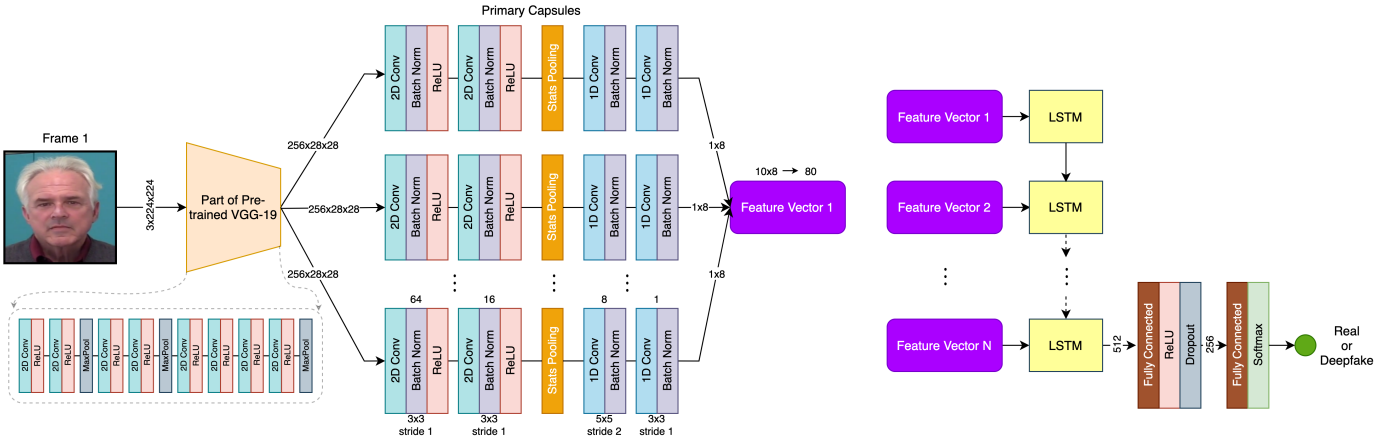


Fig. 4: Detailed Architecture: CapsuleNet + LSTM Model

XceptionNet outputs a feature vector of length 2048 which is then given as input to a single layer LSTM of 512 hidden units. The rest of the architecture after LSTM is the same as that of CapsuleNet + LSTM.

E. Performance Metrics

For evaluating our model performances, we focus on three metrics, namely:

1) Accuracy: Accuracy is the percentage of correctly classified observations i.e.

$$\text{Acc} = \frac{\text{correct}}{\text{correct} + \text{incorrect}}$$

2) Logistic Loss: Log Loss or Binary Cross Entropy measures the uncertainty of prediction based on how much it varies from the actual label. Log loss quantifies the accuracy of the model and is also used in the Kaggle competitions for evaluating models ranking users. Lower the log loss score, better the model.

$$\text{LogLoss} = -\frac{1}{n}\sum_{i=1}^{n}\left[y_i \log(\hat{y}_i) + (1-y_i)\log(1-\hat{y}_i)\right],$$

where

- n is the number of videos being predicted
- $\hat{y}_i$ is the predicted probability of the video being FAKE
- y_i is 1 if the video is FAKE, 0 if REAL
- log() is the natural (base e) logarithm

3) AUC: Area under the ROC curve is a single measure that can be used for comparing the performance of classifiers. AUC of a classifier is equal to the probability that the classifier will rank a randomly sampled positive example higher than a randomly sampled negative example. Using AUC, we can measure how much a classifier is capable of distinguishing between classes i.e. it represents the degree or measure of separability. The higher the AUC, the better the classifier in distinguishing between a real video and a deepfake video.

F. Evaluation

As we want to evaluate our models on different techniques, we use multiple validation methods. The validation steps for each of the methods can be summarized as follows:

1) Performance comparison of Convolutional Network vs Capsule Network on Single Frame: Each model is trained on a single frame taken from every video to learn the spatial features to achieve a single output score and the performance is compared using the metrics stated in Subsection III-E.

2) Performance comparison of Convolutional Network vs Capsule Network on Frame-by-Frame (Average): Each model is trained on multiple frames taken from every video to learn the spatial features to achieve multiple output scores for multiple frames and the performances are compared on the average of these scores using the metrics stated in Subsection III-E.

3) Performance comparison of Convolutional Network with LSTM vs Capsule Network with LSTM on Multiple Frames: Each model is trained on multiple frames taken from every video to learn the spatio-temporal features to achieve a single output score and the performances are compared using the metrics stated in Subsection III-E.

4) Visualization of Activation of Capsules of CapsuleNet + LSTM model: Our model is given a sequence of real and deepfake as input and the activation capsules are visualized using the open-source tool [53] implementing the Grad-CAM, Guided Backpropagation algorithm [54] and Grad x Image [55]. Using the visualizations, we can understand what each capsule focuses on, and explain how the capsules differentiate between a real and a fake video depending on the features they capture. Each capsule will learn a different feature as each capsule has a different weight initialization (initialize by using a normal distribution). For visualizations, we use the latent features extracted by the capsules before the statistical pooling layers as the 2D structure is present.

Model	Frame Selection	Accuracy	Log-Loss	AUC
CapsuleNet	Single Frame	78.49%	0.5876	0.8516
XceptionNet	Single Frame	84.48%	0.4066	0.8883
CapsuleNet	First 10 (Average)	79.36%	0.5849	0.8684
XceptionNet	First 10 (Average)	85.50%	0.3972	0.9359
CapsuleNet	Equal Interval (Average)	80.96%	0.5824	0.8996
XceptionNet	**Equal Interval (Average)**	**86.78%**	**0.3018**	**0.9571**
CapsuleNet	Most Changes (Average)	79.27%	0.5870	0.8774
XceptionNet	Most Changes (Average)	86.27%	0.3410	0.9460
CapsuleNet + LSTM	First 10	77.77%	0.5361	0.8599
XceptionNet + LSTM	First 10	83.56%	0.4236	0.9104
CapsuleNet + LSTM	**Equal Interval**	**83.42%**	**0.4763**	**0.9115**
XceptionNet + LSTM	**Equal Interval**	**85.09%**	**0.4688**	**0.9168**
CapsuleNet + LSTM	Most Changes	81.54%	0.5066	0.8873
XceptionNet + LSTM	Most Changes	84.58%	0.4752	0.9216

TABLE V: Model performance on DFDC Test Set

IV. Results

A. CapsuleNet vs XceptionNet and Single Frame vs Multiple Frames

Table V compares the performances of different models when applied to different frame selection methods. For single-frame detection, it can be seen that XceptionNet outperforms CapsuleNet by ~6% in accuracy. When an average of 10 frames is taken using the same XceptionNet and CapsuleNet models, an increase in performance in both models is seen, although a ~6% gap in accuracy remains between CapsuleNet and XceptionNet. Thus, it shows that only using a single frame is not as effective as using a sequence of frames.

Depending on the single-frame selected, there is a chance that the frame selected may be a real frame in a deepfake video and the single-frame model will classify it as a real video. In Figure 5, we use the CapsuleNet on a real video and a deepfake video to predict if every single frame is fake or not. As can be seen, the model for the deepfake video predicts some frames as real, and similarly for the real video predicts some frames as fake. Hence, it is better to consider a sequence of frames in comparison to a single frame to detect deepfake videos.

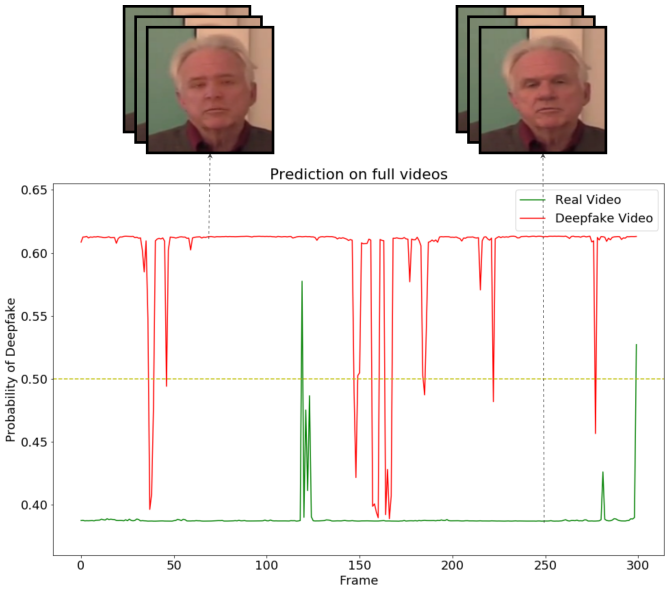


Fig. 5: Output of Capsule Network on full Real and Deepfake video.

B. Temporal Inconsistencies and CapsuleNet + LSTM

We extract the spatial features from each frame using CapsuleNet and these features are given as input to the LSTM network to find temporal inconsistencies. Based on these spatial-temporal inconsistencies, our model predicts if a given video is real or fake. From Table V, we can see the performance of using CapsuleNet + LSTM has increased the accuracy by ~5% against CapsuleNet predicting on a single frame, and ~2-2.5% against CapsuleNet predicting on average of multiple frames on the DFDC Test Set. In general, CapsuleNet + LSTM detecting spatio-temporal inconsistencies improve the model performance than using a single CapsuleNet.

When compared to the baseline XceptionNet model and XceptionNet + LSTM model, the CapsuleNet + LSTM model does not surpass the state-of-the-art model for detecting fake and real on the DFDC Test Set. However, the gap between the two is much reduced and the difference is ~3.3% in CapsuleNet + LSTM model vs XceptionNet (average of multiple frames) and ~1.7% in CapsuleNet + LSTM model vs XceptionNet + LSTM. On the other hand, XceptionNet's performance does not improve when combined with LSTM, opposite to what happened with CapsuleNet. In other words, XceptionNet alone outperforms all the other models. This may be due to the deep, heavy model and complex output of XceptionNet which fails the LSTM model to detect temporal inconsistencies between frames and does not generalize well.

On the other hand, when comparing accuracy on the Public Test Set for best performance, similar accuracy i.e. ~78% was attained by our model CapsuleNet + LSTM, state-of-the-art

XceptionNet and XceptionNet + LSTM model. Considering the Equal Interval frame selection, the XceptionNet model has a significant drop of ~7.8% in accuracy, and XceptionNet + LSTM model has a significant drop of ~6.7% in accuracy. Whereas, a smaller drop of ~5% in accuracy is seen by CapsuleNet + LSTM model. The reason for similar accuracy between both XceptionNet based models and CapsuleNet + LSTM model may be due to the augmentations applied to 79% and an additional unseen deepfake generation technique used in this data-set. This shows that our model is more robust towards unseen deepfake generation techniques and heavy augmentations than XceptionNet based models and achieves similar performance to the state-of-the-art model. As this is an open initiative by Facebook to contribute towards deepfake detection, they have shared results of top-performing models and achieves 82.56% accuracy and log-loss of 0.20336 on Public Test Set [56]. Although there is a big difference in comparison to the log-loss of our model, the difference between the accuracy is small, i.e. ~4%. As we don't have the complete Private Test Set, we cannot compare the results with the top-performing models.

When comparing accuracy on the Private Test Set for best performance, similar accuracy i.e. ~81% is achieved by both XceptionNet and XceptionNet + LSTM whereas CapsuleNet + LSTM achieves 76.64% accuracy. The reason for the significant drop in accuracy for our model in comparison to Public Test Set may be due to the additional never-seen-before filters like dog filter and flower crown filter used in Private Test Set. Although these are additional augmentations applied to the videos, tampering, and modifications are done in the face area. Hence, these videos should no more be considered as real as inconsistencies are introduced from these filters and our model classifies them as deepfakes or the filter overlaps and hide the inconsistencies introduced by deepfake generation methods and classifies them as real. Figure 6 shows examples of dog filter applied on videos and our model predicts inaccurately on most of them. Due to this reason, comparing model performance on the Private Test Set may not be accurate enough and the results are ambiguous.

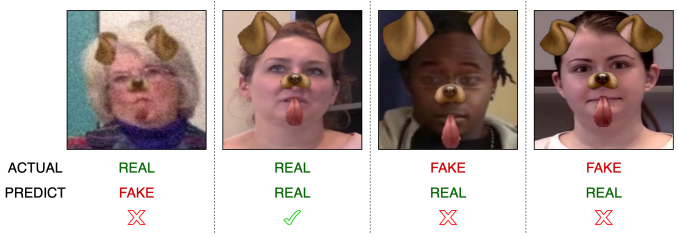


Fig. 6: Dog Filter applied on Real and Deepfake videos and their predictions.

C. Frame Selection

From Table V, we can see that the Equal Interval frame selection method always has higher performance results than the First 10 and Most Changes method in all the models. In the average of frames, the impact of frame selection method is small, i.e. ~0.1-1.6%, whereas in temporal based models, i.e. CapsuleNet + LSTM and XceptionNet + LSTM, we can see the impact is huge. Although XceptionNet + LSTM based models have a slight increase of ~1.5% from First 10 and ~0.5% from Most Changes in accuracy in comparison to Equal Interval, CapsuleNet + LSTM have a higher accuracy increase of ~5.6% from First 10 and ~1.9% from Most Changes in accuracy in comparison to Equal Interval. Similarly, when comparing the performance of frame selection methods on the Public Test Set and Private Test Set in Table VII, Equal Interval always performed better for all the models in comparison to the First 10 and Most Changes selection techniques.

The First 10 method captures the first 10 frames where the transition between two consecutive frames is captured and the difference between these two frames is least. The Most Changes method captures the transition between the most changing frames in 1-sec interval and the difference between frame i^{th} and frame $(i+1)^{th}$ is more than the First 10 method. Whereas the Equal Interval method captures 10 frames at equal intervals and the difference between frame i^{th} and frame $(i+1)^{th}$ is bigger than both the First 10 and Most Changes method and hence, the models detect higher inconsistencies when using these frames. In other words, higher differences within the selected 10 frames capture more inconsistencies and have higher performance. Therefore, the frame selection method for detecting fake videos is an important aspect and Equal Interval achieves the best performance.

Similarly, Figure 7, Figure 8 and Figure 9 shows the ROC curve using different frame selection methods for the classification of DFDC Test Set, Public Test Set and Private Test Set using the CapsuleNet + LSTM model. The area under the curve (AUC) values are shown in Table V for the DFDC Test Set and Table VII for the Public and Private Test Set. Equal Interval has the best performance, followed by Most Changes and then the First 10 method for each data-set.

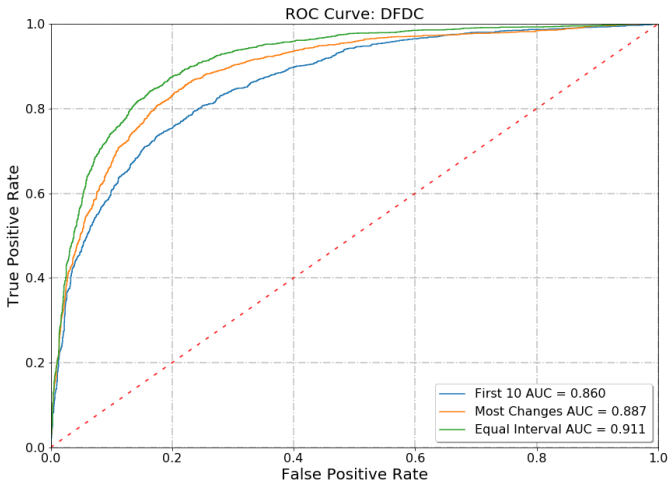


Fig. 7: ROC curve for frame selection on DFDC Test Set.

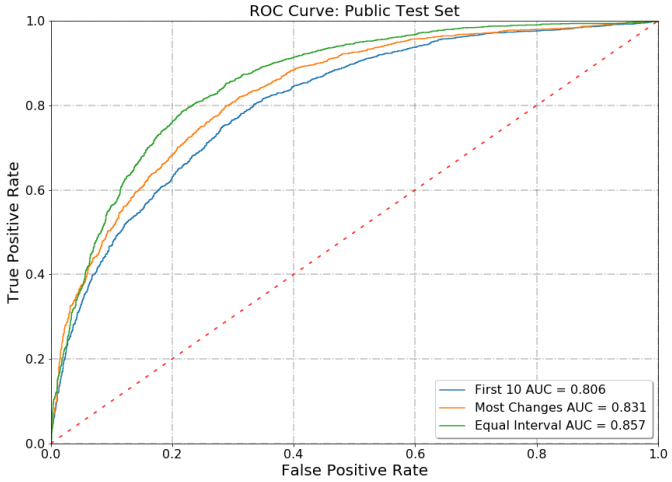


Fig. 8: ROC curve for frame selection on Public Test Set.

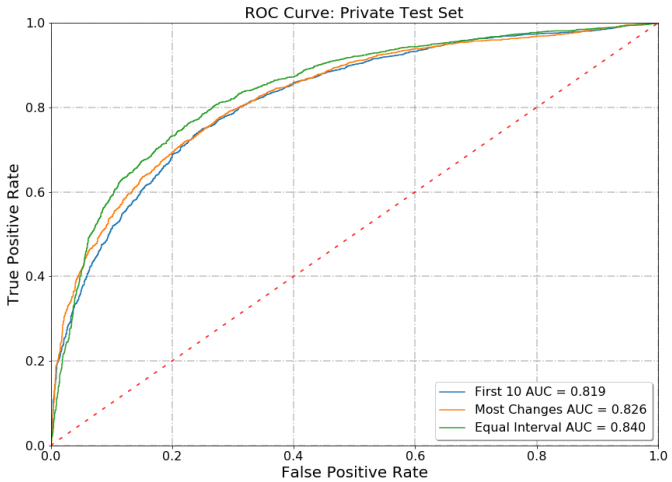


Fig. 9: ROC curve for frame selection on Private Test Set.

D. Computational Complexity

Models are trained on High-Performance Cluster using one of the multiple GPUs including Tesla P100, Titan-X, and 1080-Ti shared between multiple users. Although, XceptionNet/XceptionNet + LSTM outperforms our model CapsuleNet + LSTM with a short gap in DFDC Test Set and comparable results in Public Test Set, CapsuleNet + LSTM is much smaller than both XceptionNet and XceptionNet + LSTM. As shown in Table VI, the number of parameters of CapsuleNet + LSTM is $1/5^{th}$ of XceptionNet and $1/7^{th}$ of XceptionNet + LSTM and the size of CapsuleNet + LSTM is $1/4^{th}$ of XceptionNet and $1/5^{th}$ of XceptionNet + LSTM. Figure 10 shows a bubble graph comparing the model size on the x-axis, accuracy on the y-axis, and the number of parameters as the area of the bubble. The accuracy here is for the Equal Interval frame selection method on the DFDC Test Set. Therefore, when comparing our CapsuleNet + LSTM model to state-of-the-art XceptionNet model, for ~3% drop in accuracy in DFDC Test Set, our model is 4 times smaller in size and 5 times smaller in the number of parameters, hence making it lighter and reduced computational cost than the state-of-the-art model. It will require fewer resources and power to be used in distributed systems or integrating into online social media platforms for real-time identification of deepfakes at a lower computational cost. Hence, focusing more on state-of-the-art efficiency than state-of-the-art accuracy.

Model	No. of Parameters (in million)	Size (in MB)
CapsuleNet	2.79M	6.3MB
CapsuleNet + LSTM	4.03M	21.1MB
XceptionNet	21.86M	87.8MB
XceptionNet + LSTM	27.24M	109.3MB

TABLE VI: Model Size

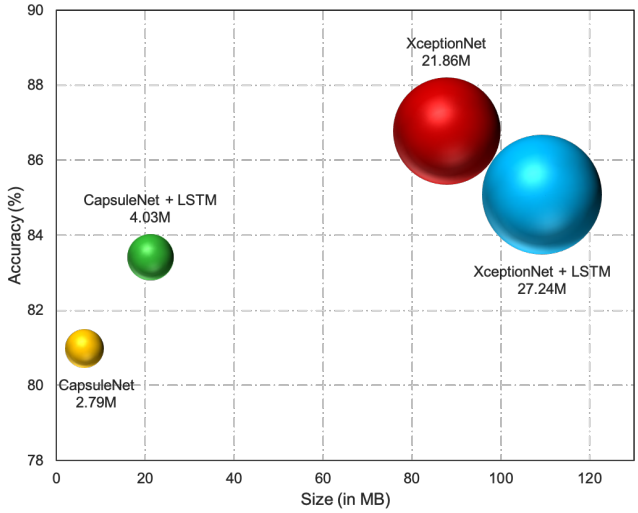


Fig. 10: Computational comparison of the Models. The X-axis is the Size of the Models, Y-axis is the Accuracy for the Frame-selection "Equal Interval" method in the DFDC Test Set and the Area of the Bubble is the Number of Parameters.

E. Visualization of Capsules and Spatial Features Extracted

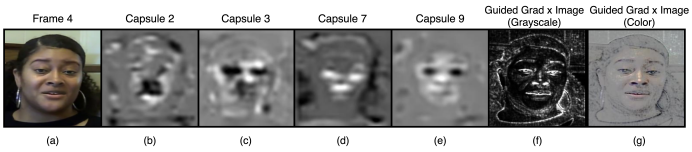
To understand how our model predicts a given video as real or fake, we use the open-source tool [53] implementing the Grad-CAM, Guided Back-propagation algorithm [54] and Grad x Image [55], we visualize the capsules of CapsuleNet + LSTM model for a real video and a fake video as shown in Figure 11, we can see the input image (a) and their respective visualizations (b)-(g). Figure (b)-(e) are the Gradient-weighted Class Activation Map (Grad-CAM) of different capsules, where each capsule is identified as a different feature in the input image. In Figure 11i, (b) focuses below the mouth region, (c) focuses outside the eyes and nose region, (d) focuses on eyes and nose, (e) focuses on the whole face excluding eyes. (f) and (g) are the output of Guided Grad X Image in grayscale and color. It can be seen that the capsules mostly focus on the facial regions of the whole face when identifying the given sequence of frames as fake.

Model	Frame Selection	Public Test			Private Test		
		Accuracy	Log-Loss	AUC	Accuracy	Log-Loss	AUC
CapsuleNet	Single Frame	71.65%	0.6188	0.7837	72.19%	0.6151	0.7969
XceptionNet	Single Frame	75.50%	0.5651	0.8153	77.21%	0.5396	0.8287
CapsuleNet	First 10 (Average)	71.63%	0.6170	0.7997	73.49%	0.6084	0.8213
XceptionNet	First 10 (Average)	76.83%	0.5918	0.8674	78.81%	0.5622	0.8930
CapsuleNet	Equal Interval (Average)	73.63%	0.6129	0.8241	75.88%	0.6041	0.8480
XceptionNet	**Equal Interval (Average)**	**78.99%**	**0.4979**	**0.8863**	**81.08%**	**0.4552**	**0.9170**
CapsuleNet	Most Changes (Average)	72.28%	0.6166	0.8096	74.51%	0.6075	0.8305
XceptionNet	Most Changes (Average)	77.34%	0.5369	0.8744	79.74%	0.4929	0.9049
CapsuleNet + LSTM	First 10	72.73%	0.6271	0.8059	74.61%	0.6128	0.8195
XceptionNet + LSTM	First 10	77.75%	0.5356	0.8576	78.20%	0.5270	0.8625
CapsuleNet + LSTM	**Equal Interval**	**78.38%**	**0.5699**	**0.8567**	**76.64%**	**0.6624**	**0.8403**
XceptionNet + LSTM	**Equal Interval**	**78.38%**	**0.5489**	**0.8744**	**81.16%**	**0.4993**	**0.8831**
CapsuleNet + LSTM	Most Changes	74.67%	0.6140	0.8308	74.53%	0.6143	0.8263
XceptionNet + LSTM	Most Changes	77.44%	0.5345	0.8659	80.52%	0.4915	0.8843

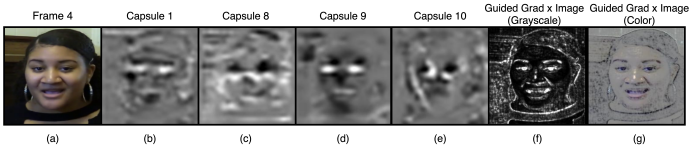
TABLE VII: Model performance on Public Test Set and Private Test Set

In Figure 11ii, (b) focuses below the eyes region, (c) focuses on the lower face, (d) focuses on the eyes, (e) focuses on the region around the eyes. (f) and (g) are the output of Guided Grad X Image in grayscale and color. It can be seen that the capsules mostly focus on facial regions around the eyes, nose, and mouth when identifying the given sequence of frames as real.

Hence, the model's predictions are based on different facial regions of the face for both real and fake videos. Most capsules focus on facial areas like eyes, nose, mouth, and regions around them, some capsules miss to detect these facial regions and some fail to detect the manipulated regions. However, with multiple capsules, these features are collected, combined, and captured as spatial features. Using these spatial features detected in each frame and combining them across multiple frames to get temporal features, the model detects inconsistencies across spatial and temporal domains. Therefore, predicting a given video as real or fake.



(i) Fake frame



(ii) Real frame

Fig. 11: Capsule visualizations of CapsuleNet + LSTM model for a real and fake frame. (a) is the input mage. (b)-(e) are the Grad-CAM vis of different capsules, each capsule learn a different feature. (f) and (g) are the output of Guided Grad X Image in grayscale and color.

V. Conclusions

With the ongoing rise in Deepfake videos, this paper provides an exhaustive survey of the recent advancements in fake media generation and their threat towards authenticity and trustworthiness of online information. We describe the deepfake generation methods and the inconsistencies introduced by the AI, available data-sets, and how current deepfake detection techniques exploit these inconsistencies.

This paper provides a new spatio-temporal hybrid model using CapsuleNet integrated with LSTM. CapsuleNet is used to identify spatial inconsistencies from a single frame that are introduced by the computer while generating deepfake videos. Using Grad-CAM and Guided Back-propagation, we visualize the activation of capsules when a fake video and a real video are given as input. From these visualizations, we can see that each capsule learns a different facial feature and focuses mostly on facial regions like eyes, outside eyes, nose, mouth, and whole face excluding eyes for both real and fake videos. Hence, Capsule Network extracts these facial features and combines to create a feature vector which consists of spatial inconsistencies in the frame. These feature vectors consisting of spatial inconsistencies are collected for 10 frames and then given as input to our LSTM model to identify temporal inconsistencies within these frames and predict whether the video, from which these 10 frames are extracted is real or fake. CapsuleNet alone by just detecting spatial inconsistencies achieves good performance with an accuracy score of 80.98% (average of multiple frames) but fails to outperform the state-of-the-art model XceptionNet by the accuracy of ~5.8% on DFDC Test Set.

CapsuleNet when integrated with LSTM to detect both spatial and temporal inconsistencies improves the performance by ~2.5% and achieves an accuracy score of 83.42% on the DFDC Test Set. On comparing with the XceptionNet model and XceptionNet integrated with the LSTM model, our model although fails to outperform both the state-of-the-art model

by ~3.3% and state-of-the-art model integrated with LSTM by ~1.7%, the difference between them is almost comparable in DFDC Test Set. When comparing the performance on Public Test Set, our model, XceptionNet and Xception integrated with LSTM have similar accuracy of ~78%, the significant drop in XceptionNet based models possibly be due to heavy augmentations and a new never-seen-before deepfake generation technique in Public Test Set, whereas our model has a smaller drop of ~5% showing CapsuleNet integrated with LSTM is more robust towards new generation techniques and augmentations. On comparison with the top-performing models shared by Facebook, both state-of-the-art XceptionNet and CapsuleNet+LSTM lacks behind by ~4% on Public Test Set. When considering the model size and the number of parameters, Capsule Network integrated with LSTM (~4M parameters) is much smaller than XceptionNet (~22M parameters), almost $1/5^{th}$ and XceptionNet integrated with LSTM (~27M parameters), almost $1/7^{th}$. Therefore, CapsuleNet integrated with LSTM can achieve comparable accuracy with the advantage of being a lighter model and hence, reduced computational cost, focusing more on being state-of-the-art efficiency than state-of-the-art accuracy. Hence, the model requires fewer resources and power and is more suitable to be used in distributed systems and online social media platforms, like Facebook and Instagram, to classify an uploaded video as real or deepfake in real-time and watermark them to avoid the spread of fake videos.

We also see that the frame selection method has a significant impact on the performance in every model. CapsuleNet integrated with the LSTM model when using the Equal Interval method improved by ~5.65% when compared to the First 10 method and ~1.88% when compared to the Most Changes method on DFDC Test Set. In other words, Equal Interval achieves the best performance followed by Most Changes and First 10 and is also seen for Public Test Set and Private Test Set. The reason behind this increase in performance is probably due to the differences between multiple frames is highest in Equal Interval followed by Most Changes and is least in First 10. Therefore, higher differences capture higher inconsistencies across multiple frames improve the detection of fake videos. Therefore, a frame selection technique does have a significant impact on the detection of fake videos, and the Equal Interval method achieves the best results.

With these promising results, we can finally answer our research question: **"Can the performance of detecting inconsistencies in a video to identify deepfakes be improved by combining a long short-term memory network with a capsule network to create a spatio-temporal hybrid model?"** CapsuleNet integrated with LSTM creates a spatio-temporal hybrid model that improves the performance of deepfake detection in videos in comparison to CapsuleNet and achieves comparable results to state-of-the-art XceptionNet while being a lighter model and having the number of parameters $1/5^{th}$ of XceptionNet. Future work could include the ensemble of models, heavy augmentations, and attention mechanisms for the improvement of deepfake detection.

REFERENCES

[1] "People are using face-swapping tech to add Nicolas Cage to random movies and what is 2018." [Online]. Available: https://mashable.com/2018/01/31/nicolas-cage-face-swapping-deepfakes/?europe=true

[2] "A deepfake artist's attempt to make Robert De Niro look younger in 'The Irishman' is being hailed as superior to Netflix's CGI." [Online]. Available: https://www.businessinsider.nl/deepfake-netflix-correcting-the-irishman-de-ageing-tech-2020-1?international=true&r=US

[3] "Deepfake Detection Challenge - YouTube." [Online]. Available: https://www.youtube.com/watch?v=wxsijKZRSCQ

[4] "You Won't Believe What Obama Says In This Video! - YouTube." [Online]. Available: https://www.youtube.com/watch?v=cQ54GDm1eL0

[5] "Deepfakes by BJP in Indian Delhi Election Campaign - VICE." [Online]. Available: https://www.vice.com/en_in/article/jgedjb/the-first-use-of-deepfakes-in-indian-election-by-bjp

[6] "A Voice Deepfake Was Used To Scam A CEO Out Of $243,000." [Online]. Available: https://www.forbes.com/sites/jessedamiani/2019/09/03/a-voice-deepfake-was-used-to-scam-a-ceo-out-of-243000/#11031c382241

[7] "Pelosi videos manipulated to make her appear drunk are being shared on social media - YouTube." [Online]. Available: https://www.youtube.com/watch?v=sDOo5nDJwgA

[8] "FakeApp 2.2.0 - Download for PC Free." [Online]. Available: https://www.malavida.com/en/soft/fakeapp/#gref

[9] "deepfakes/faceswap: Deepfakes Software For All." [Online]. Available: https://github.com/deepfakes/faceswap

[10] "shaoanlu/faceswap-GAN: A denoising autoencoder + adversarial losses and attention mechanisms for face swapping." [Online]. Available: https://github.com/shaoanlu/faceswap-GAN

[11] "iperov/DeepFaceLab: DeepFaceLab is the leading software for creating deepfakes." [Online]. Available: https://github.com/iperov/DeepFaceLab

[12] "dfaker/df: Larger resolution face masked, weirdly warped, deepfake,." [Online]. Available: https://github.com/dfaker/df

[13] J. Thies, M. Zollhöfer, M. Stamminger, C. Theobalt, and M. Nießner, "Face2Face: real-time face capture and reenactment of RGB videos," *Communications of the ACM*, vol. 62, no. 1, pp. 96–104, 2018. [Online]. Available: https://dl.acm.org/doi/10.1145/3292039

[14] S. Suwajanakorn, S. M. Seitz, and I. Kemelmacher-Shlizerman, "Synthesizing obama: Learning lip sync from audio," in *ACM Transactions on Graphics*, vol. 36, no. 4. Association for Computing Machinery, 2017, pp. 1–13. [Online]. Available: https://dl.acm.org/doi/10.1145/3072959.3073640

[15] "The Irishman De-Aging: Netflix Millions VS. Free Software! - YouTube." [Online]. Available: https://www.youtube.com/watch?v=dyRvbFhknRc

[16] X. Yang, Y. Li, and S. Lyu, "Exposing Deep Fakes Using Inconsistent Head Poses," in *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, vol. 2019-May. IEEE, 2019, pp. 8261–8265. [Online]. Available: https://ieeexplore.ieee.org/document/8683164/

[17] P. Korshunov and S. Marcel, "DeepFakes: a New Threat to Face Recognition? Assessment and Detection," pp. 1–5, 2018. [Online]. Available: http://arxiv.org/abs/1812.08685

[18] A. Rössler, D. Cozzolino, L. Verdoliva, C. Riess, J. Thies, and M. Nießner, "FaceForensics++: Learning to Detect Manipulated Facial Images," 2019. [Online]. Available: http://arxiv.org/abs/1901.08971

[19] "Google AI Blog: Contributing Data to Deepfake Detection Research." [Online]. Available: https://ai.googleblog.com/2019/09/contributing-data-to-deepfake-detection.html

[20] Y. Li, X. Yang, P. Sun, H. Qi, and S. Lyu, "Celeb-DF: A Large-scale Challenging Dataset for DeepFake Forensics," 2019. [Online]. Available: http://arxiv.org/abs/1909.12962

[21] B. Dolhansky, R. Howes, B. Pflaum, N. Baram, and C. C. Ferrer, "The Deepfake Detection Challenge (DFDC) Preview Dataset," 2019. [Online]. Available: http://arxiv.org/abs/1910.08854

[22] "Join the Deepfake Detection Challenge (DFDC)." [Online]. Available: https://deepfakedetectionchallenge.ai/

[23] B. Dolhansky, J. Bitton, B. Pflaum, J. Lu, R. Howes, M. Wang, and C. C. Ferrer, "The DeepFake Detection Challenge Dataset," 2020. [Online]. Available: http://arxiv.org/abs/2006.07397

[24] "Deepfake Detection Challenge — Kaggle." [Online]. Available: https://www.kaggle.com/c/deepfake-detection-challenge

[25] Y. Li, M.-C. Chang, and S. Lyu, "In Ictu Oculi: Exposing AI Generated Fake Face Videos by Detecting Eye Blinking," *2018 IEEE International Workshop on Information Forensics and Security (WIFS)*, pp. 1–7, 2018. [Online]. Available: https://ieeexplore.ieee.org/document/8630787/

[26] P. Zhou, X. Han, V. I. Morariu, and L. S. Davis, "Two-Stream Neural Networks for Tampered Face Detection," 2018. [Online]. Available: http://arxiv.org/abs/1803.11276

[27] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going Deeper with Convolutions," 2014. [Online]. Available: http://arxiv.org/abs/1409.4842

[28] D. Afchar, V. Nozick, J. Yamagishi, and I. Echizen, "MesoNet: a Compact Facial Video Forgery Detection Network," in *2018 IEEE International Workshop on Information Forensics and Security (WIFS)*. IEEE, 2018, pp. 1–7. [Online]. Available: https://ieeexplore.ieee.org/document/8630761/

[29] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi, "Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning," 2016. [Online]. Available: http://arxiv.org/abs/1602.07261

[30] D. Guera and E. J. Delp, "Deepfake Video Detection Using Recurrent Neural Networks," in *2018 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*. IEEE, 2018, pp. 1–6. [Online]. Available: https://ieeexplore.ieee.org/document/8639163/

[31] Y. Li and S. Lyu, "Exposing DeepFake Videos By Detecting Face Warping Artifacts," 2018. [Online]. Available: http://arxiv.org/abs/1811.00656

[32] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2016-Decem, 2016, pp. 770–778. [Online]. Available: http://arxiv.org/abs/1512.03385

[33] F. Chollet, "Xception: Deep Learning with Depthwise Separable Convolutions," *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, vol. 2017-Janua, pp. 1800–1807, 2016. [Online]. Available: http://arxiv.org/abs/1610.02357

[34] F. Matern, C. Riess, and M. Stamminger, "Exploiting Visual Artifacts to Expose Deepfakes and Face Manipulations," in *2019 IEEE Winter Applications of Computer Vision Workshops (WACVW)*. IEEE, 2019, pp. 83–92. [Online]. Available: https://ieeexplore.ieee.org/document/8638330/

[35] H. H. Nguyen, F. Fang, J. Yamagishi, and I. Echizen, "Multi-task Learning For Detecting and Segmenting Manipulated Facial Images and Videos," 2019. [Online]. Available: http://arxiv.org/abs/1906.06876

[36] E. Sabir, J. Cheng, A. Jaiswal, W. AbdAlmageed, I. Masi, and P. Natarajan, "Recurrent Convolutional Strategies for Face Manipulation Detection in Videos," pp. 80–87, 2019. [Online]. Available: http://arxiv.org/abs/1905.00582

[37] H. H. Nguyen, J. Yamagishi, and I. Echizen, "Use of a Capsule Network to Detect Fake Images and Videos," 2019. [Online]. Available: http://arxiv.org/abs/1910.12467

[38] S. Sabour, N. Frosst, and G. E. Hinton, "Dynamic routing between capsules," in *Advances in Neural Information Processing Systems*, vol. 2017-Decem, 2017, pp. 3857–3867. [Online]. Available: http://arxiv.org/abs/1710.09829

[39] H. H. Nguyen, J. Yamagishi, and I. Echizen, "Capsule-Forensics: Using Capsule Networks to Detect Forged Images and Videos," 2018. [Online]. Available: http://arxiv.org/abs/1810.11215

[40] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," 2014. [Online]. Available: http://arxiv.org/abs/1409.1556

[41] S. Agarwal, H. Farid, Y. Gu, M. He, K. Nagano, and H. Li, "Protecting world leaders against deep fakes," *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pp. 38–45, 2019. [Online]. Available: http://openaccess.thecvf.com/content_CVPRW_2019/papers/MediaForensics/Agarwal_Protecting_World_Leaders_Against_Deep_Fakes_CVPRW_2019_paper.pdf

[42] G. E. Hinton, A. Krizhevsky, and S. D. Wang, "Transforming Auto-Encoders," 2011, pp. 44–51. [Online]. Available: http://link.springer.com/10.1007/978-3-642-21735-7_6

[43] G. Hinton, S. Sabour, and N. Frosst, "Matrix capsules with EM routing," in *6th International Conference on Learning Representations, ICLR 2018 - Conference Track Proceedings*, 2018.

[44] T. Iesmantas and R. Alzbutas, "Convolutional Capsule Network for Classification of Breast Cancer Histology Images," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 10882 LNCS, 2018, pp. 853–860.

[45] R. Saqur and S. Vivona, "CapsGAN: Using Dynamic Routing for Generative Adversarial Networks," in *Advances in Intelligent Systems and Computing*, 2020, vol. 944, no. Nips, pp. 511–525. [Online]. Available: http://link.springer.com/10.1007/978-3-030-17798-0_41

[46] K. Duarte, Y. S. Rawat, and M. Shah, "CapsuleVOS: Semi-Supervised Video Object Segmentation Using Capsule Routing," 2019. [Online]. Available: http://arxiv.org/abs/1910.00132

[47] ——, "VideoCapsuleNet: A Simplified Network for Action Detection," 2018. [Online]. Available: http://arxiv.org/abs/1805.08162

[48] "yeephycho/tensorflow-face-detection: A mobilenet SSD based face detector, powered by tensorflow object detection api, trained by WIDERFACE dataset." [Online]. Available: https://github.com/yeephycho/tensorflow-face-detection

[49] A. Buslaev, V. I. Iglovikov, E. Khvedchenya, A. Parinov, M. Druzhinin, and A. A. Kalinin, "Albumentations: Fast and Flexible Image Augmentations," *Information*, vol. 11, no. 2, p. 125, 2020. [Online]. Available: https://www.mdpi.com/2078-2489/11/2/125

[50] Y. Yao, W. Hu, W. Zhang, T. Wu, and Y.-Q. Shi, "Distinguishing Computer-generated Graphics from Natural Images Based on Sensor Pattern Noise and Deep Learning," 2018. [Online]. Available: http://dx.doi.org/10.3390/s18041296

[51] H. H. Nguyen, T. N.-D. Tieu, H.-Q. Nguyen-Son, V. Nozick, J. Yamagishi, and I. Echizen, "Modular Convolutional Neural Network for Discriminating between Computer-Generated Images and Photographic Images," in *Proceedings of the 13th International Conference on Availability, Reliability and Security - ARES 2018*. New York, New York, USA: ACM Press, 2018, pp. 1–10. [Online]. Available: http://dl.acm.org/citation.cfm?doid=3230833.3230863

[52] I. Loshchilov and F. Hutter, "Decoupled Weight Decay Regularization," *7th International Conference on Learning Representations, ICLR 2019*, 2017. [Online]. Available: http://arxiv.org/abs/1711.05101

[53] U. Ozbulak, "PyTorch CNN Visualizations," 2019. [Online]. Available: https://github.com/utkuozbulak/pytorch-cnn-visualizations

[54] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization," 2016. [Online]. Available: http://dx.doi.org/10.1007/s11263-019-01228-7

[55] A. Shrikumar, P. Greenside, A. Shcherbina, and A. Kundaje, "Not Just a Black Box: Learning Important Features Through Propagating Activation Differences," 2016. [Online]. Available: http://arxiv.org/abs/1605.01713

[56] "Deepfake Detection Challenge Results: An open initiative to advance AI." [Online]. Available: https://ai.facebook.com/blog/deepfake-detection-challenge-results-an-open-initiative-to-advance-ai/