Signal, Image and Video Processing (2024) 18:6613–6629
https://doi.org/10.1007/s11760-024-03339-2

ORIGINAL PAPER

Lightweight 3D-StudentNet for defending against face replay attacks

Preethi Jayappa Seegehalli[1] · B. Niranjana Krupa[1]

Received: 7 November 2023 / Revised: 14 May 2024 / Accepted: 31 May 2024 / Published online: 13 June 2024

Abstract
Biometric face and lip-reading systems are susceptible to face replay attacks. Where the intruder presents a recorded video of a legitimate user or presents printed photos to gain system access without authorization. Consequently, liveness detection becomes essential to confirm whether the person in camera view is real rather than fake replays. This study aims to detect such printed photo attacks and video or photo replay attacks on high-resolution screens. The main objective of this research is to develop a lightweight 3D-DNN model that considers both spatial and temporal features to distinguish between real and attack videos. To achieve this objective, a lightweight 3D-StudentNet face replay attack defense system is proposed leveraging 3D-ArrowNet deep neural network and knowledge distillation. The system captures dynamic spatio-temporal features from five video frames captured by an RGB camera. Considerable experimentation is conducted using the Replay-Attack and Replay-Mobile benchmark datasets. Experimental results demonstrate that the proposed 3D-ArrowNet achieves state-of-the-art performance and transfers its knowledge successfully to a lightweight 3D-StudentNet with fewer network parameters. Thus, 3D-StudentNet can supplant existing 2D-DNN architectures utilized for replay attack defense systems, which capture only spatial features.

Keywords Face biometrics · Lip-reading biometrics · Presentation attack · Face replay attack · Liveness detection · Deep neural network · Knowledge distillation

1 Introduction

Text-based passwords for login systems are prone to attacks like man-in-behind attacks, where passwords can be stolen by look-over or man-behind; brute force attacks, where, the attacker attempts several digit combinations until the secret four digit password is obtained; spoofing attacks, where attackers present closely resembling data to enter the system, offline guessing attack, where intruder try guessing the passwords until the secret password is verified, and user impersonation attack where other users will use authorized client credentials to log in. Also, a text-based password does not guarantee the liveness of a legitimate user. Thus, Face recognition and lip-password recognition are becoming popular biometric approaches. However, they are at risk of face replay attacks. In case of a video attack to hack the system, legitimate users' recorded video is replayed in the view of the RGB camera. On the other hand, a photo attack involves displaying a legitimate user's photo in the view of an RGB camera to gain access to the system. Unlike face biometric systems, lip-password biometric systems are effective against photo attacks, as lip movements are used as passwords. Despite illumination and camera variations, face replay attack defense systems must derive discriminative features to differentiate between real and fake videos. Moire patterns and reflections are prevalent patterns observed in video and photo attacks. For print attacks, color distortion is a powerful spoof pattern because prints have a narrow color dynamic range compared to captured real faces. Also, temporal motion frames in real and fake videos convey the spoof cues like changes in texture and color distortions. Attack videos exhibit inconsistent patterns for skin textures. If the camera captures faces from presented photographs or videos, the clarity of facial features is degraded, creating discontinuities in reflectivity. However, the greatest challenge arises from the developments in the camera and display technologies, which can make face-replay attack videos and photos closely resemble real face videos.

This study focuses primarily on photo and video attacks for face and lip-reading biometrics. The static face replay

✉ Preethi Jayappa Seegehalli
preethisj@pes.edu

[1] Department of ECE, PES University, Bangalore 560085, India

attack defense system (FRADS) only leverages the spatial features of a video frame. In contrast, dynamic FRADS considers both spatial and temporal features. In the literature, even though the performance of 3D-DNNs (which consider both spatial and temporal features from video frames) is superior to 2D-DNNs for other video classification applications like human action recognition, medical imaging [1], visual speech recognition [2], their application in face replay attack defense systems remains unexplored due to the increased number of trainable network parameters and storage space requirements.

The current systems designed to defend against face replay attacks typically use 2D-VGG16, 2D-ResNets, 2D-DenseNets, and 2D-MobileNets as their backbone architecture. However, these systems only take into account spatial features and do not consider temporal features or features extracted from various filter sizes. The 2D-Inception network uses different filter sizes ($1 \times 1, 3 \times 3$, and 5×5), but it does not include skip connections like ResNet or DenseNet, which can improve the model's performance. To consider temporal features, 3D convolutions are necessary. The kernel or filter dimension of 3D convolutions in the 3D convolutional layer is an essential factor to consider. When it comes to capturing minute details in images, smaller kernel dimensions such as $1 \times 1 \times 1$, $3 \times 3 \times 3$, or $5 \times 5 \times 5$ are preferred as they do not learn from far neighboring pixels. Conversely, larger kernel sizes like $7 \times 7 \times 7$ and $11 \times 11 \times 11$ capture more general features as they take into account far neighboring pixels. However, selecting the optimal kernel dimension is challenging as it captures different details and is more specific to a particular application.

The first objective of this research is to create a tailored and efficient 3D-DNN model that can tackle the shortcomings of the current 2D-DNN backbone architectures utilized for detecting face replay attacks. To achieve this, a customized 3D-ArrowNet is designed, which combines Inception's idea of using different kernel size filters and DenseNet's idea of skip connections and depth concatenation. The designed 3D-ArrowNet consists of four streams, where each stream uses different 3D-convolutional kernel sizes to learn the spoof features. Like DenseNet, the architecture employs skip connection and depth concatenation ideas to improve the model's performance. Although using multiple streams and large kernel size filters increases computational complexity, it results in the best performance.

Since 3D-ArrowNet is computationally intensive, the second objective of this research is to design a lightweight 3D-DNN model by reducing the computational complexity of 3D-ArrowNet architecture. To achieve this objective, the 3D-StudentNet is designed with low computational complexity by employing techniques like stream pruning, filter pruning, and replacing the large kernel size filter with a series of smaller kernel size filters. However, this leads to a decrease in the performance of 3D-StudentNet. Further, Knowledge distillation (KD) is leveraged to address this issue, where a trained 3D-ArrowNet is used as a teacher to train a 3D-StudentNet model, which helps to significantly improve the performance of 3D-StudentNet to match its teacher's accuracy.

Thus, this work's principal contributions are summarized as follows: In contrast to other Deep Learning (DL) based approaches, the proposed method embeds HSV face video frames as input to the DL model instead of RGB video frames input. The hue and saturation planes of the HSV color space specify the chrominance, and the value plane specifies the brightness. HSV color space is appropriate for discriminating the color textures between real and attack face videos. The proposed work is centered on developing a lightweight 3D deep neural network (DNN) model for a dynamic face replay attack defense system that considers both spatio-temporal features. The novel aspect of this investigation is the 3D-ArrowNet architecture design, whose computational complexity is greater but has superior performance in detecting the attack and real videos. It first learns to classify the attack and real videos and then teaches the same to the 3D-StudentNet model by transferring its knowledge via knowledge distillation. The proposed 3D-ArrowNet and 3D-StudentNet framework is evaluated using the popular benchmarks "Replay-Mobile" and "Replay-attack". Along with 3D-ArrowNet and 3D-StudentNet, a comprehensive experimental analysis of static backbone architectures such as 2D-VGG16, 2D-ResNet18, 2D-DenseNet201, and 2D-MobileNet-V2 is carried out. The results reveal that 3D-StudentNet performs better in preventing face replay attacks with fewer trainable network parameters and can, therefore, supplant the backbone architectures of current replay attack defense systems. The outcomes of the models trained with the combined Replay-Mobile and Replay-Attack datasets are also presented. The models are trained and tested on the combined dataset to generate more robust and accurate models than those trained on the individual dataset. A combined dataset has more subjects, camera types, display screen types, printer types, and recording environments, making it more challenging than a single dataset.

The remaining sections are structured as follows: Sect. 2 reviews related works. Section 3 introduces the pipeline of the Face replay attack defense system. Section 4 describes the experimental outcomes, and Sect. 5 compares the results with state-of-the-art techniques on Replay Attack and Replay Mobile datasets. Finally, Sect. 6 concludes the paper and discusses some prospective research directions.

2 Related works

In light of their remarkable success in other applications, researchers have investigated the potential of pattern recognition, machine learning, and DL strategies in developing a Face replay attack defense system (FRADS). A progressive improvement in the employed methodologies is observed over time. Handcrafted features with machine learning (ML) techniques were used in the early classification era. In the current era, DL and hybrid DL approaches are investigated. The following paragraph describes several hybrid DL strategies used for developing static FRADS.

The VGG-Face Real-Fake classifier, Principal Component Analysis (PCA), and Support Vector Machine (SVM) was leveraged for classification [3]. Single Shot MultiBox Detectors (SSDs) with VGG16 backbones was used to recognize faces with confidence scores from images. Then, SVM classified Spatial Pyramid Coding Micro-Texture features if the confidence score was below the threshold [4]. VGG-Face was trained for Real-Spoof class detection. Then, the SVM classifier was used to classify the local binary pattern (LBP) features extracted from the middle convolutional layers' convolutional feature maps [5]. Face images were utilized to derive LBP and simplified Weber local descriptor (SWLD) features. Then, the SVM classifier was used to classify cascaded LBP-CaffeNet-Face and SWLD-CaffeNet-Face features [6]. Faster R-CNN was utilized to determine the confidence score of face detection. In cases where the confidence score was found to be less than 0.9, enhanced Retinex-based LBP features were extracted, and an SVM classifier was employed to calculate the face detection score. The final decision was based on thresholding the average of the scores obtained from both. FARCNN was enhanced using an attention-based fusion technique and numerous loss functions, such as Regression loss, Crystal loss, and Center loss [7]. The subsequent paragraph provides the literature that has employed various hybrid DL strategies for dynamic FRADS.

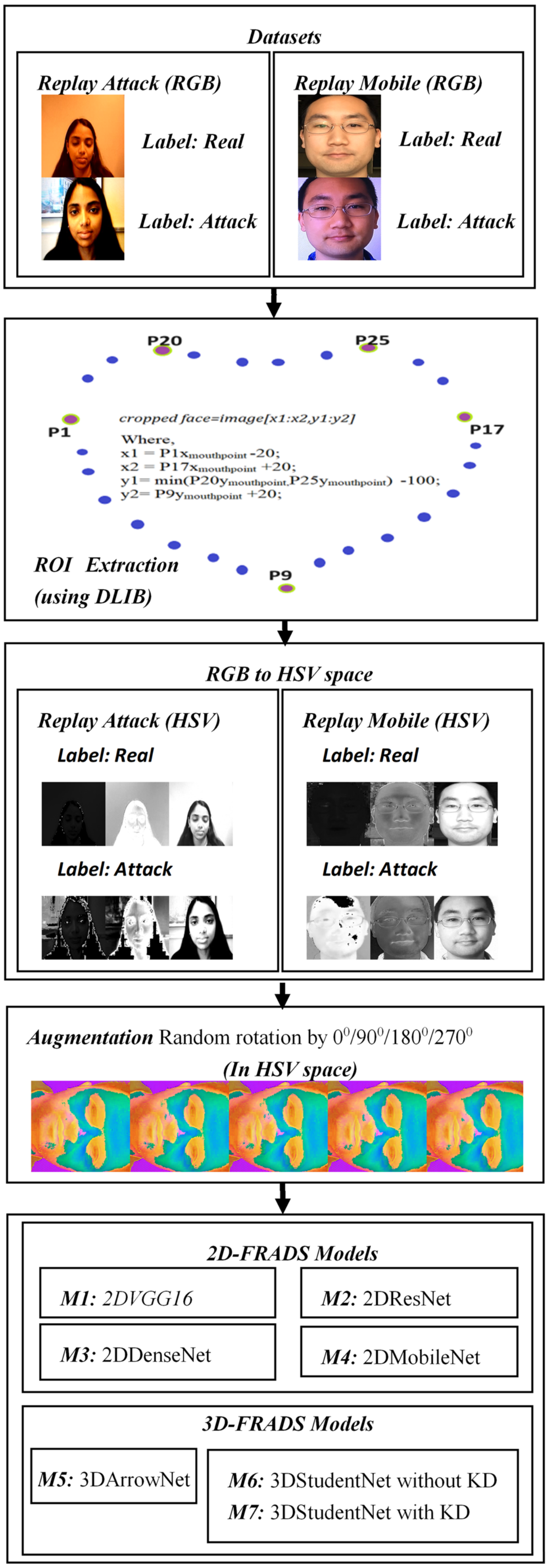
Face image quality was assessed using the Shearlet transform, while motion-based face replay attack features were extracted using dense optical flow. These features were combined using a hierarchical neural network bottleneck feature fusion approach [8]. Five-layer 2D-CNN was used to extract the video frame's features. The third, fourth, and fifth convolutional layers' generated image sequences were saved to construct LBP-TOP histograms, which were then concatenated and normalized to train the SVM classifier [9]. Motion blur was magnified using Eulerian motion magnification (EMM). To extract blur-intensity variation features, a 1D-CNN network was customized to process cascaded intensity histograms of video frames. Furthermore, a histogram based on local similar patterns (LSP) was used to extract blur-width features. The extracted features were then combined (either early or late) and classified using SVM [10].

DL strategies have been proven to be highly effective even without using handcrafted features. Several scholarly works employ 2D-DL networks trained with cross-entropy loss for static FRADS. Hyperopt-convnet architecture optimization and filter optimization was leveraged to build a two-layer CNN [11]. Transfer learning of VGG16 [12], VGG-11 and its modified networks were employed for classification [13]. The full face and four face regions were used to train a hybrid CNN, and the features extracted by the final fully connected (FC) layers of the hybrid CNN were concatenated and classified by SVM [14]. A depth map was generated from an RGB video frame to distinguish real and attack video frames [15]. Another study utilized multi-scale deep and shallow AlexNet architectures with long short-term memory (LSTM). The network was designed to receive a face Region of Interest (ROI) scaled at three levels. The LSTM algorithm was used to combine the features from multiple scales, and all the features were fused based on the weights predicted by LSTM [16]. The face region was divided into nine patches. Nine Caffe-network PatchNets learned to classify these patches during the initial phase. Full facial images were used to fine-tune the model during the second phase [17]. Bootstrapping technique with modified VGG11 network was leveraged [18]. RGB, HSV, and YCbCr color spaces were investigated for FRADS. Three convolution and two FC layers were proposed, and a fusion strategy based on majority voting was ultimately chosen [19]. Seven-layered CNN FRADS used auto-encoder-generated and real-world face images. The first two Siamese Conv layers shared weights to reduce disparity in feature maps. Then, the weighted fusion layer generated feature maps of the previous Conv layers for training [20]. Contrastive loss was used to train a Siamese network with a modified Alexnet architecture backbone [21]. A self-supervised Regional Fully Convolution Network (SSR-FCN) with two training phases, first with a full-face image and second with random-sized patches from attack regions, was proposed [22]. The face replay attack defense system performs poorly during cross-camera training and testing. As a solution, researchers have developed a two-stream network that includes a camera-invariant stream, designed with a ResNet18-based pseudo-Siamese network, and a feature discrimination stream. The latter extracts features from enhanced images that are constructed from high and low-frequency image components. During training, binary focal loss and camera type loss were used [23]. Researchers combined NUAA, CASIA, and Replay Attack datasets for training so that aggregated train sets offered diverse attack variations. Pretrained VGG-16, ResNet-50, Inception V3, and DenseNet-121 were employed for transfer learning. The trained models were tested on an unseen SiW dataset [24]. The attack and defense systems can be considered analogous to generators and discriminators. The attack system consisted of two parts: Adv-APG and Sup-APD. First, Adv-APG translated a live face into a spoofing sample that contained known attack features. Then, Sup-APD drifted the spoofing sample into the agnostic domain. The discriminator (any FRADS backbone) classified both real and spoof faces and made the generator generate agnostic attacks with unseen features [25]. In another work, a two-stream framework was used with an EfficientFormerV2-S2 backbone. The framework accepted RGB and Gaussian bandpass-filtered image inputs in the two input streams. The two stream features were combined using a cross-attention fusion architecture, and training was optimized using the lion optimizer [26].

Few scholarly works have worked on DL networks trained with cross-entropy loss for constructing dynamic FRADS. For anti-spoofing, a 2D-CNN+LSTM approach was used to differentiate between real and attack videos by extracting spatial and temporal features [27]. The binary classification was carried out using ResNet50+LSTM [28]. The filmed face is not a rigid plane in actual face videos. Hence, the frames are not related by homography transformation. The replaying screen or attacking photo is a rigid plane, so all frames are related by homography transformation. Therefore, the Local Motion Discovery Module (LMDM) extracted local motion features from 9 image patches, which were classified using 2D-CNN [29]. The Spatio-Temporal Anti-Spoofing Network (STASN) was developed with three modules. First, the Temporal Anti-Spoofing Module (TASM) used Res-Net50 followed by LSTM to identify the temporal dependencies among video frames. Second, the Spatial Anti-Spoofing Module (SASM) received input from the Region Attention Module (RAM), which utilized GradCam to provide substantial local region input. Third, SASM was comprised of K streams, from which discriminative features were extracted by sharing their weights [30]. ResNet-50 derived features were normalized and augmented using sparse filtering and then fed to an LSTM [31]. A vision-based video transformer was proposed as the backbone for detecting presentation attacks using a multi-scale, multi-head, and self-attention framework [32]. Previous research on FRADS focused on developing static models that utilized 2D-CNNs to analyze the spatial features of a single RGB video frame. Some works on dynamic FRADS have analyzed multiple RGB video frames to account for both spatial and temporal features, but they have either utilized 2D-CNNs with LSTM or computationally intensive vision transformers. To address this limitation, a 3D-CNN with fewer network parameters is designed for building dynamic FRADS.

3 Proposed methodology

The pipeline of proposed methodology is presented in Fig. 1.



cropped face=image[x1:x2,y1:y2]
Where,
$x1 = P1x_{mouthpoint} -20;$
$x2 = P17x_{mouthpoint} +20;$
$y1= \min(P20y_{mouthpoint,}P25y_{mouthpoint}) -100;$
$y2= P9y_{mouthpoint} +20;$

Fig. 1 Pipeline of Proposed Methodology

This study examines the efficiency of the proposed 3D frameworks “3D-ArrowNet (M5)” and lightweight student framework “3D-StudentNet with knowledge distillation (M7)” by comparing its performance to that of four 2D backbone architectures commonly used for static FRADS: 2D-VGG16 (M1) [33], 2D-ResNet18 (M2) [34], 2D-DenseNet201 (M3) [35], and 2D-MobileNet-V2 (M4) [36]. The models are tested using the Replay-Attack, Replay-Mobile, and combined (Replay-Mobile+Replay-Attack) datasets.

3.1 Dataset

For all experiments, Replay-Attack (RA), Replay-Mobile (RM), and combined datasets (RA+RM) are utilized. Training set, validation set (or development set), and test set are already provided with the datasets. The total number of video samples utilized in the experiments for training, validation, and testing is depicted in Table 1. Replay-Attack has more number of attack samples and less number of real samples as compared to Replay-Mobile. Samples from both databases are combined to form a combined dataset.

3.1.1 Replay-attack (RA)

Replay-Attack [37] dataset comprises 1200 video recordings (320 × 240) of real accesses and fake accesses of 50 persons recorded using two types of cameras 12.1MP Canon-PowerShot-SX150-IS and with a 3.1MP iPhone-3GS. The videos were captured under two setups: (a) Controlled setup: with a uniform background and artificial lighting conditions; (b) Adverse setup: with a non-uniform background and natural lighting conditions. Three attacks and two supports have been considered. Printed, mobile, and high-resolution attacks are three categories of attacks with two support types: fixed and handheld. In the print attack, a Triumph-Adler DCC 2520 color laser printer was used to produce high-resolution images on A4 paper, which were then shown to the camera. During the mobile attack, legitimate persons’ photographs and videos captured by the iPhone were displayed on the iPhone’s screen and then shown to the camera. During the high-resolution attack, legitimate persons’ high-quality photographs and videos captured by the HQ camera were displayed on the iPad screen (1024 × 768) and then shown to the camera. Attack videos were recorded using two distinct support types. In Fixed support, the attack source was fixed using a support, whereas in handheld, the attacker held the attack source in their hands. The Replay-Attack dataset has three subsets, with the first subset containing 15 individuals for training, the second subset containing 15 individuals for validation, and the third subset containing 20 individuals for testing. Replay attack videos are 10 s long and have a frame rate of 25 fps.

3.1.2 Replay-mobile (RM)

Replay-Mobile [38] dataset includes 1190 video recordings (1280 × 720) of real and fake accesses of 40 people. The recordings were made using two types of cameras: Nikon Coolpix P520 with an 18MP and LG-G4 mobile 1080p HD. In the real access videos, each person recorded ten videos under five illumination conditions, which included controlled, adverse, direct, lateral, and diffuse lighting. They used two types of devices: tablet and mobile. The attack videos were captured under two different illumination conditions: with and without lights on. There were two types of attack: matte screen and print. In the matte screen attacks, the videos or photos of legitimate users were displayed on a 1920 × 1080 monitor and captured using capturing devices. All matte screen attacks were fixed, with a stand holding the capturing device. In the print attacks, the subjects’ faces were printed on A4 matte paper and presented to a handheld or fixed capturing device. The Replay-Mobile dataset has three subsets: the first subset includes 12 individuals for training, the second consists of 16 individuals for validation, and the third contains 12 individuals for testing. Replay mobile videos are 10 s long and have a frame rate of around 30 fps.

3.2 Preprocessing

The Replay-Mobile and Replay-Attack datasets consist of 10 s video recordings captured at 30 fps and 25 fps, respectively. As a result, each video in the Replay-Mobile dataset contains around 300 frames, while each video in the Replay-Attack dataset contains around 250 frames. Utilizing all frames for training and testing 3D-FRADS models will significantly increase the computational complexity of the network, making it unsuitable for real-time applications. Therefore, experiments are conducted on the Replay-Mobile dataset to determine the optimal number of video frames to input into the 3D-ArrowNet model. As discussed in Sect. 3.3.2, the input with five frames is chosen as it is optimal regarding computational time and performance.

Preprocessing consists of three steps: first, the extraction of face ROI from the first five video frames, then the conversion of frames from RGB to HSV space, and finally, the random rotation of all frames by 0^0/ 90^0/ 180^0/ 270^0. Following this, the first frame is fed into 2D FRADS models, while the first five frames are input into 3D FRADS models.

Face ROIs are extracted from first five video frames using “Shape Predictor 68 Face Landmarks” trained weights and the DLIB [39] library as depicted in Fig. 1. The landmarks points between the numbers P_1 and P_{68} indicate the face region. The face is cropped using minimum and maximum x-coordinates and y-coordinates of the key face points (P_1, P_9, P_{17}, P_{20}, and P_{25}). The ROI is cropped by adding

Table 1 The number of video samples employed for train, validation and test

Dataset	#of Train video samples		#of Validation video samples		#of Test video samples	
	#Real	#Attack	#Real	#Attack	#Real	#Attack
RA	60	300	60	300	80	400
RM	120	192	160	256	110	192
RA+RM	180	492	220	556	190	592

and subtracting some pixels from the image's ROI maximum and minimum coordinate values, respectively to include the small amount of background region. Then, the cropped face is resized to [250, 250] using Lanczos3 interpolation to suppress aliasing artifacts. For each video, the first five frames of cropped face in RGB domain are converted to HSV domain, a more suitable domain space for detecting color distortion. Pixel intensities contain real and fake patterns, and modifying them may corrupt them. Therefore, no augmentations are carried out that manipulate their intensities. The only augmentation technique used is rotating all the video frames together by 0^0, 90^0, 180^0, or 270^0 because rotating the frames does not affect their intensities or alter their label from real to fake or vice versa. This makes the model concentrate on identifying spoof patterns rather than face structure.

3.3 FRADS models

3.3.1 2D-FRADS models

The performance of the designed 3D-FRADS backbone models is compared with 2D-FRADS backbone models: 2D-VGG16 (M1) [33], 2D-ResNet18 (M2) [34], 2D-DenseNet201 (M3) [35], and 2D-MobileNet-V2 (M4) [36]. The VGG16 framework includes 13 2D-conv layers, 5 max-pools, and 3 FC layers. Due to the use of FC layers and an increasing number of filters, it is computationally expensive. The 18-layer 2D-ResNet18 framework consists of 17 2D-Conv layers and 1 FC layer. In a vanilla CNN, the output of each layer flows directly into the next layer. In ResNet, however, there is a residual connection where the output of one layer is added to the input of the following layer, allowing the data to skip some of the layers in between. Batch normalization (BN) and rectified linear unit function (ReLu) were introduced in ResNet, DenseNet, and MobileNet.

2D-DenseNet201 is 201 layers deep. It comprises the dense blocks and transition blocks. The next dense block receives the feature embeddings of the previous dense block and is concatenated depth-wise. Mobile applications utilize MobileNetV2, which are 53 layers deep. Its framework has an inverted residual structure. The number of channels in a vanilla Residual Block employs a wide-to-narrow and narrow-to-wide flow. First, the input has more channels, and 1 × 1 2D-Conv is used to compress the channel numbers, and then again, it is increased using 1 × 1 2D-Conv so that input and output can be added. Conversely, an Inverted Residual Block employs a narrow-to-wide and wide-to-narrow flow. First, the input has fewer channels and 1 × 1 2D-Conv is used to expand the number of channels and then uses a 3 × 3 lightweight depth-wise separable 2D-convolutions to filter features and to introduce nonlinearity. Again, this is followed by a 1 × 1 2D-Conv to decrease the channel numbers so that input and output can be added. MobileNetV2's architecture consists of an initial 2D-Conv layer with 32 filters, followed by 19 bottleneck layers.

2D-VGG [3–5, 12, 13, 18, 40–42], 2D-ResNet [23, 24, 28, 30, 31, 43–46], 2D-DenseNet [24, 47], and 2D-MobileNet [48] are employed as 2D-FRADS backbone models in the literature. They all have more trainable parameters to fetch deep features. Even though MobileNet backbones need fewer Flops than other 2D backbones, their performance is not as good as deeper networks with more Flops. Because all 2DCNNs take a single video frame as input, they inherently fail to leverage context from adjacent video frames, thus capturing focus on spatial features and can't capture temporal features. Thus, to capture both spatio-temporal features along with 2D models, LSTMs are tried in [16, 27, 28, 30, 31, 41]. 2DCNNs extract spatial features, whereas LSTMs are good at the long-term temporal learning of 2DCNNs' extracted spatial features. However, LSTMs have a complex architecture and are susceptible to gradient explosions. Visual Transformer and self-attention networks are explored in literature but they are computationally intensive. In FRADS literature, 3DCNNs, which are capable of capturing spatial and temporal features, are less explored. 3DCNNs are computationally more expensive than 2DCNNs. As a result, knowledge distillation was employed in the work to reduce the computational complexity.

3.3.2 3D-ArrowNet

3D-Convolutions are utilized to build the 3D-ArrowNet architecture. The size of the kernel is an important parameter in the 3D convolutional layer. Kernels with dimensions of 1 × 1 × 1, 3 × 3 × 3, or 5 × 5 × 5 are more effective in capturing fine details, while larger kernels like 7 × 7 × 7 and

11 × 11 × 11 are better suited for detecting general features. However, choosing the right kernel dimension can be challenging. Thus, to overcome this issue, the model employs multiple streams that capture details from different kernel dimensions. Combining features from different streams and selecting the best ones that suit the application through training can help improve the model's overall performance.

Several experiments are conducted on the Replay Mobile dataset with five frames video input, to determine the optimal number of streams and kernel sizes for capturing essential spoofing patterns. The results are depicted in Table 2. Upon analyzing the results of the experiments, it was determined that the optimal architecture would include four distinct streams with four different kernel sizes: 11 × 11 × 11, 7 × 7 × 7, 5 × 5 × 5, and 3 × 3 × 3. Thus, the features are extracted through four different streams with four different kernel sizes and then concatenated depth-wise. Concatenated features are further processed by the transition block and the tail of the 3D-ArrowNet to extract the best features.

The experiments are also conducted on the Replay-Mobile dataset to determine the optimal number of frames for the 3D-ArrowNet model input layer. The first layer of the 3D-ArrowNet model is the video input layer, which accepts N video frames of size 250 × 250 × 3. During the experiment, N was tested with different numbers of video input frames, namely one, three, five, seven, nine, and eleven. The results of the experiment are presented in Table 3. The results indicate that employing seven, nine, and eleven frames as input requires more training time and is impractical for real-time applications. Whereas 3D-ArrowNet with five input frames show better results compared to other numbers of input frames with an accuracy of 99.66%. Therefore, in the experiments, N is set to five frames.

The proposed 3D-ArrowNet architecture depicted in Fig. 2, has 72 layers and 1.3 M learnable network parameters with nineteen 3D-convolutional layers and one FC /dense layer with two nodes. The architecture employs depth concatenation, which helps the successive layer to receive the previous layers' feature embeddings. As depicted in the figure, the 3D-ArrowNet's video input layer can take in a sequence of five video frames, each with a size of 250 × 250 pixels and three color channels (HSV).

The head of the ArrowNet consists of 4 streams, where each stream uses different convolutional kernel sizes to learn the spoof patterns. The first stream use kernel size of 11 × 11 × 11, the second stream use kernel size of 7 × 7 × 7, the third stream use kernel size of 5 × 5 × 5, and the fourth steam use kernel size of 3 × 3 × 3 in the first 3D-conv layer. Second stream contains [3D-conv with 64 filters, BN, ReLu, 3 × 3 × 3 max pool, BN, ReLu, 1 × 1x1 3D-conv with 128 filters, BN, ReLu, and 3 × 3 × 3 3D-conv with 32 filters], whereas the other three streams contain "3 × 3 × 3 3D-conv with 32 filters" instead of "1 × 1 × 1 3D-conv with 128 filters". The features learned by all streams, along with the features derived from the 3 × 3 × 3 max pool layer of the second stream, are depth-wise concatenated and forwarded to the tail. CNN features learned by [11 × 11 × 11, 7 × 7 × 7, 5 × 5 × 5, and 3 × 3 × 3] convolution streams are depicted in Fig. 3. The image on the left depicts channel-1 features, while the image on the right depicts channel-2 features. It is evident from the feature maps that the 11 × 11 × 11 kernel stream captured general features while the 3 × 3 × 3 kernel stream captured minute details. The four streams are followed by a transition block (TB), which connects the head and tail of the ArrowNet. TB block consists of [BN, ReLu, 1 × 1 × 1 3D-Conv, and 2 × 2 × 2 Avg Pool].

Transition block reduces the size of the video by half. The tail of the ArrowNet consists of two consecutive dense blocks (DB). Each DB block consists of [BN, ReLu, 1 × 1 × 1 3D-Conv, BN, ReLu, 3 × 3 × 3 3D-Conv with 32 filters] layers followed by depth concatenation. This is then followed by [BN, ReLu, 1 × 1 × 1 3D-Conv with 256 filters, BN, ReLu, 1 × 1 × 1 3D-Conv with 256 filters, 2 × 2 × 2 Avg Pool, BN, ReLu]. A ReLU layer is an activating function defined by f(x) = max (0, x). ReLU speeds up the training of a network by omitting insignificant features of an image. Hence improves the model's performance. Layers of Batch Normalization (BN) are added for input normalization and regularization. It helps to speed up the convergence of training time. BN speeds up training by decreasing the number of iterations required to attain the desired loss value. Using the specified number of filters, a 1 × 1 × 1 3D-Conv modifies the dimension of the channel to the desired value. The spatial and temporal dimensions are cut in half by 3D-conv with stride 2. In addition, average pooling with a stride of two reduces the spatial and temporal dimensions by half. The network's final output is 256 features, and the last three layers consist of 3D global average pooling (GAP), a dense layer, and a softmax layer that generates probability scores for the two output classes, real and attack. This framework can generate features that consider both spatial and temporal dimensions. Table 4 lists the mini-batch size, network input size, output features size, computational cost (Network's learnable parameters), total number of layers present in each framework, depth of each framework and the complexity of network in terms of Flops. Each network is designed to accept input data of a fixed dimension. After cropping the face region, input can be made to suit network input dimensions by resizing the input to the desired network input size.

According to the results presented in Table 5, the 3D-ArrowNet model has shown superior performance to other 2D-FRADS models. It achieved 100% accuracy, 0 FAR, 0 FRR, and 0 ACER on the Replay-Attack dataset. On the Replay-Mobile dataset, it scored an accuracy of 99.66%, 0 FAR, 0.9 FRR, and an ACER of 0.45. Also, the model has shown good performance on a challenging combined dataset

Table 2 The accuracy of the model structures tried during 3D-ArrowNet design (with input five frames)

Model	Single stream with 7 × 7 × 7 kernel dimension	Two streams with 3 × 3 × 3 and 7 × 7 × 7 kernel dimension	Three streams with 3 × 3 × 3, 7 × 7 × 7, and 9 × 9 × 9 kernel dimension	Four streams with 3 × 3 × 3, 7 × 7 × 7, 9 × 9 × 9, and 11 × 11 × 11 kernel dimension	Four streams with 3 × 3 × 3, 5 × 5 × 5, 7 × 7 × 7, and 11 × 11 × 11 kernel dimension
Accuracy (%)	98.01	98.01	99.00	99.00	99.66

Table 3 Trails carried out to select the optimal number of frames to input into the 3D-ArrowNet model

# input frames	1	3	5	7	9	11
Accuracy (%)	98.67	98.01	99.66	99.00	98	99.33
Training time per epoch (min)	4.9	9.49	13.69	18.68	24.75	33.5

with an accuracy of 99.23%, FAR of 0.1689, FRR of 2.6316, and an ACER of 1.4.

However, due to 11 × 11 × 11, 7 × 7 × 7, and 5 × 5 × 5 kernel size filters, the computational complexity of the designed 3D-ArrowNet architecture is 12G Flops; consequently, the 3D-StudentNet architecture is designed to reduce the intensive computations.

3.3.3 3D-StudentNet

The 3D-StudentNet architecture is designed with a series of 3 × 3 × 3 kernel size filters instead of large kernel size filters. It was observed experimentally that two 3 × 3 × 3 convolution layers applied successively give the same effect as a single 5 × 5 × 5 convolution layer. The latter requires many Flops, whereas the former requires fewer Flops [33].

Compared to 3D-ArrowNet, in 3D-StudentNet architecture, computation complexity is further reduced by employing a single stream instead of four streams and by reducing the total filters utilized in the head from 64 to 16 as depicted in Fig. 4. Experiments on the Replay Mobile, Replay Attack, and combined datasets, as depicted in Table 6, reveal that pruning layers and reducing the total number of streams and filters decreased 3D-StudentNet's performance. So, in order to enhance the performance of 3D-StudentNet, 3D-ArrowNet is used as its teacher to impart its knowledge on 3D-StudentNet using knowledge distillation (KD) [49]. The pipeline of knowledge distillation is illustrated in Fig. 5.

Both the teacher and student networks receive identical video training inputs and target labels and are trained with knowledge distillation (KD) loss; however, the teacher's network parameters are not updated during training, and only the network parameters of the student are modified to suppress KD loss. "Logits" refer to the un-normalized outputs of a DNN layer. The softmax function is employed as a normalization technique in deep learning to bring the "Logits" between 0 and 1 while ensuring that their sum remains equal to 1.

If "L_i" represents the "Logits" of a DNN layer, then softmax operation with temperature (T) is defined as depicted in Eq. (1).

$$p(L_i, T) = \frac{\exp(L_i/T)}{\sum_i \exp(L_i/T)} \tag{1}$$

T is equal to 1 for hard output and T is equal to a value greater than 1 for soft output. In the experiment, temperature (T) was assigned a value 10 through trial and error. The cross entropy loss is depicted in Eq. (2).

$$\text{Crossentropy} = -\sum_i t_i \log(\, p(L_i, T)) \tag{2}$$

where t_i represents the i^{th} element of the one-hot label vector t (for Loss_hard) or the i^{th} element from the Soft_output of the teacher network (for Loss_soft). The dark knowledge contained in the soft outputs of a trained 3D-ArrowNet model will be conveyed to the student network. Thus, Loss_soft represents the cross-entropy loss between 3D-ArrowNet and 3D-StudentNet's soft outputs taken from the 3D-Global average pooling (GAP) layer, which outputs a feature vector of size 256. Loss_hard represents the cross-entropy loss between the hard output from the 3D-StudentNet taken from its softmax layer with T = 1 and a one-hot ground truth label. Using Eq. (3), knowledge distillation (KD) loss is computed.

$$\text{KD Loss} = \text{Loss\_soft x } T^2 + \text{ Loss\_hard} \tag{3}$$

Fig. 2 Architecture of 3D-ArrowNet

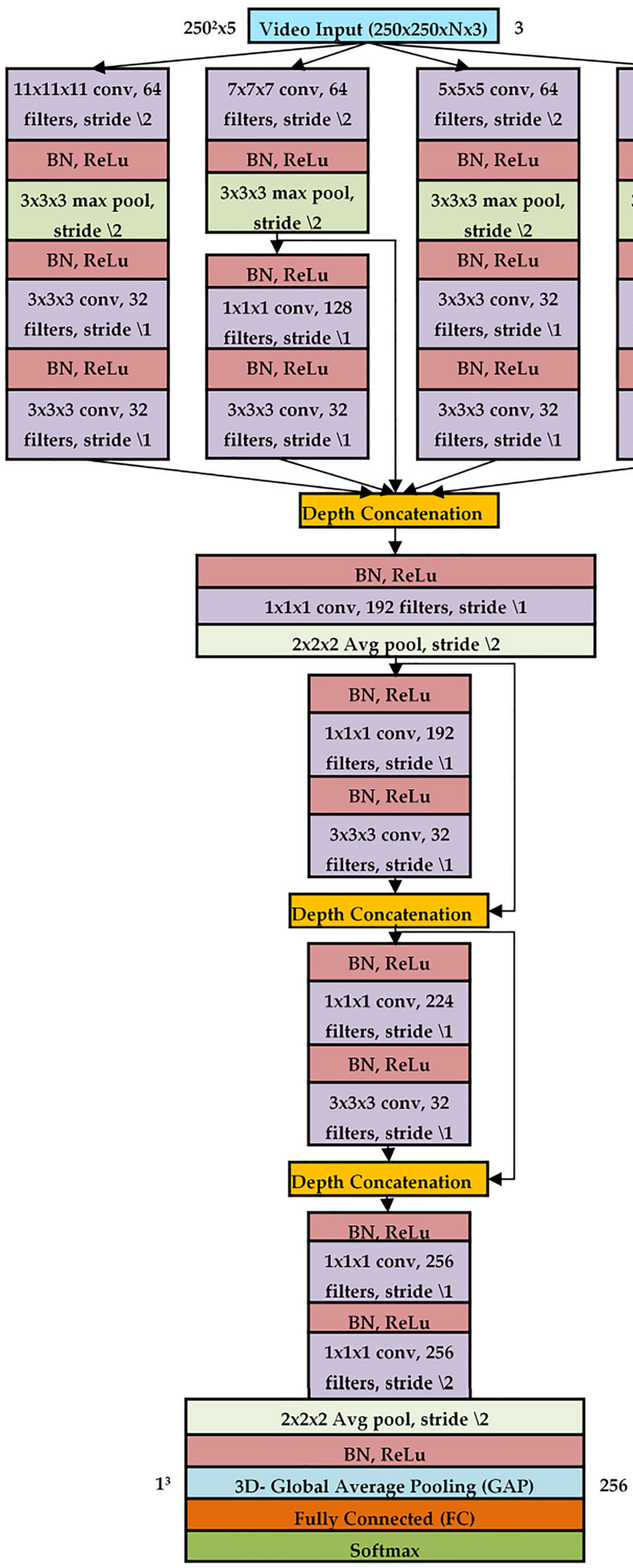

250²x5
Video Input (250x250xNx3)
3
11x11x11 conv, 64 filters, stride \2
BN, ReLu
3x3x3 max pool, stride \2
BN, ReLu
3x3x3 conv, 32 filters, stride \1
BN, ReLu
3x3x3 conv, 32 filters, stride \1
7x7x7 conv, 64 filters, stride \2
BN, ReLu
3x3x3 max pool, stride \2
BN, ReLu
1x1x1 conv, 128 filters, stride \1
BN, ReLu
3x3x3 conv, 32 filters, stride \1
5x5x5 conv, 64 filters, stride \2
BN, ReLu
3x3x3 max pool, stride \2
BN, ReLu
3x3x3 conv, 32 filters, stride \1
BN, ReLu
3x3x3 conv, 32 filters, stride \1
3x3x3 conv, 64 filters, stride \2
BN, ReLu
3x3x3 max pool, stride \2
BN, ReLu
3x3x3 conv, 32 filters, stride \1
BN, ReLu
3x3x3 conv, 32 filters, stride \1
Depth Concatenation
BN, ReLu
1x1x1 conv, 192 filters, stride \1
2x2x2 Avg pool, stride \2
BN, ReLu
1x1x1 conv, 192 filters, stride \1
BN, ReLu
3x3x3 conv, 32 filters, stride \1
Depth Concatenation
BN, ReLu
1x1x1 conv, 224 filters, stride \1
BN, ReLu
3x3x3 conv, 32 filters, stride \1
Depth Concatenation
BN, ReLu
1x1x1 conv, 256 filters, stride \1
BN, ReLu
1x1x1 conv, 256 filters, stride \2
2x2x2 Avg pool, stride \2
BN, ReLu
1³
3D- Global Average Pooling (GAP)
256
Fully Connected (FC)
Softmax

Table 4 Framework's computational complexity

Frame work	M1: 2D-VGG16	M2: 2D-ResNet18	M3: 2D-DenseNet201	M4: 2D-MobileNet	M5: 3D-ArrowNet	M6 & M7: 3D-StudentNet
Mini batch	16	128	16	128	32	32
Input size	224 × 224 × 3	224 × 224 × 3	224 × 224 × 3	224 × 224 × 3	250 × 250 × N × 3	250 × 250 × N × 3
Output features	4096	512	1920	1280	256	256
# Parameters (M)	134.2	11.1	18.1	2.2	1.3	0.692
# layers	41	71	708	154	72	50
# Depth	16	18	201	53	20	14
# Flops (G)	15.3	1.8	4	0.3	12	2.5

Table 5 The outcomes of five experimental models (M1-M5)

Model	Dataset	Accuracy (%)	FAR (%)	FRR (%)	ACER (%)
M1: 2D-VGG16	RA	97.7083	0	13.75	6.8750
	RM	97.351	2.6042	2.7273	2.6658
	Combined	91.3043	7.4324	12.6316	10.0320
M2: 2D-ResNet18	RA	98.333	0	10	5
	RM	96.0265	0.8446	6.3636	3.6041
	Combined	96.5473	1.3514	10	5.6757
M3: 2D-DenseNet201	RA	99.7917	0.2500	0	0.1250
	RM	98.0132	1.0417	3.6364	2.3391
	Combined	99.3606	0.3378	1.5789	0.9584
M4: 2D-MobileNet –V2	RA	95.8333	0.5000	22.5000	11.5000
	RM	98.0132	2.6042	0.9091	1.7567
	Combined	93.8619	1.8581	19.4737	10.6659
M5: 3D-ArrowNet (proposed)	RA	100	0	0	0
	RM	99.6689	0	0.9091	0.4545
	Combined	99.2327	0.1689	2.6316	1.4002

Table 6 The outcomes of 3D-StudentNet (Before and After Knowledge distillation)

Model	Dataset	Accuracy (%)	FAR (%)	FRR (%)	ACER (%)
M6: 3D-StudentNet (Before KD)	RA	97.08	3.25	1.25	2.25
	RM	98.67	0	3.636	1.81
	Combined	98.85	0.68	2.63	1.655
M7: 3D-StudentNet (After KD)	RA	100	0	0	0
	RM	99.6689	0	0.9091	0.454
	Combined	98.98	0.51	2.63	1.57

4 Experimental results

MATLAB deep learning toolbox and computer vision toolbox are leveraged to implement and train all models on the NVIDIA GeForce GTX 1050 Ti GPU with training being optimized using stochastic gradient descent (SGDM) algorithm with a momentum of 0.9, L2 regularization of 0.0001, and a learning rate equal to 0.001. Training checkpoints are saved after every epoch. Finally, the training checkpoint with the maximum training and validation accuracy is considered. The results achieved using 3D-ArrowNet (Dynamic FRADS), a model that is built and trained from scratch is compared with the results obtained using Static FRADS models, 2D-VGG16, 2D-ResNet18, 2D-DenseNet201, and

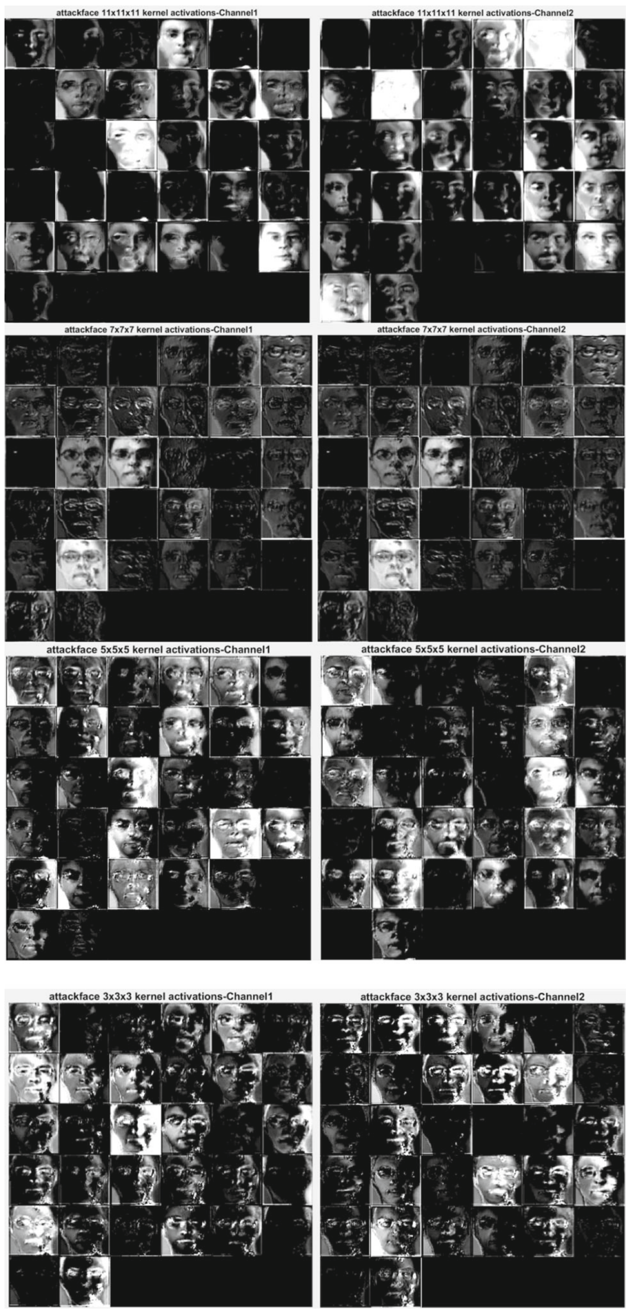

attackface 11x11x11 kernel activations-Channel1
attackface 11x11x11 kernel activations-Channel2
attackface 7x7x7 kernel activations-Channel1
attackface 7x7x7 kernel activations-Channel2
attackface 5x5x5 kernel activations-Channel1
attackface 5x5x5 kernel activations-Channel2
attackface 3x3x3 kernel activations-Channel1
attackface 3x3x3 kernel activations-Channel2

Fig. 3 CNN features learned by [$11 \times 11 \times 11$, $7 \times 7 \times 7$, $5 \times 5 \times 5$, and $3 \times 3 \times 3$] convolution streams

2D-MobileNet-V2. Static FRADS models are obtained using transfer learning. Transfer learning is performed by removing the final three layers and introducing a new FC layer, softmax layer, and classification layer, with two classes (Real and attack).

Results are analyzed using three datasets: RA, RM and combined datasets. False Acceptance (FA) refers to a situation in which a presented video is an “Attack”, but the system classifies it as “Real”. False Acceptance rate (FAR) refers

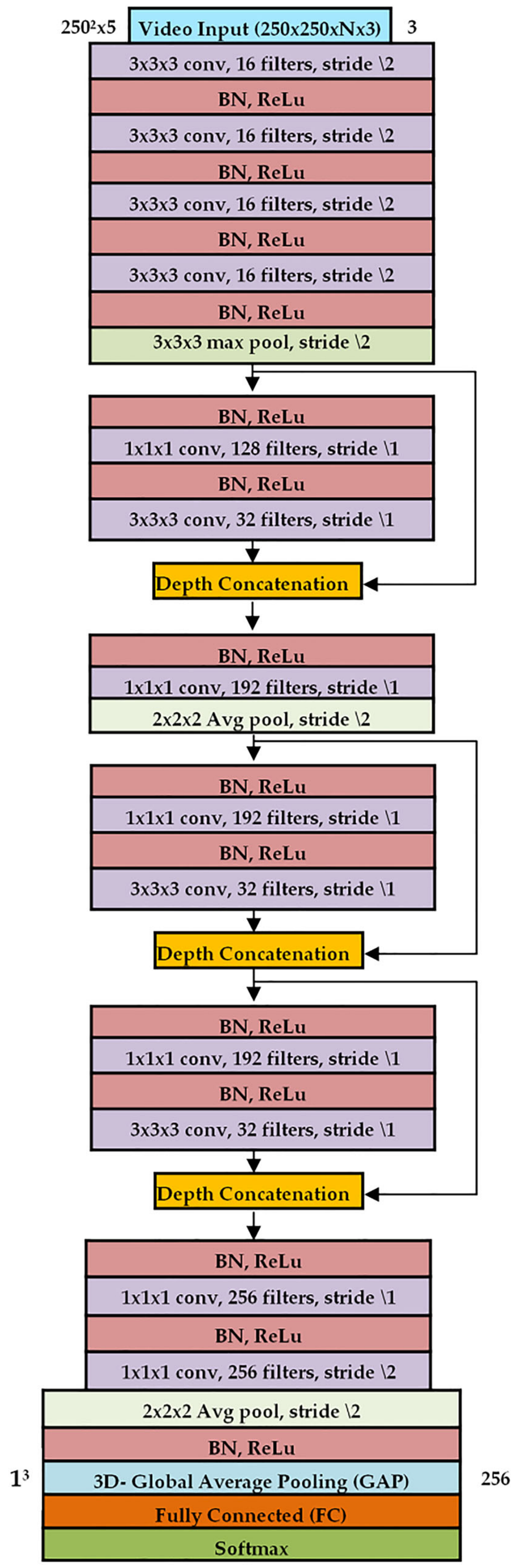

250²x5
Video Input (250x250xNx3)
3
3x3x3 conv, 16 filters, stride \2
BN, ReLu
3x3x3 conv, 16 filters, stride \2
BN, ReLu
3x3x3 conv, 16 filters, stride \2
BN, ReLu
3x3x3 conv, 16 filters, stride \2
BN, ReLu
3x3x3 max pool, stride \2
BN, ReLu
1x1x1 conv, 128 filters, stride \1
BN, ReLu
3x3x3 conv, 32 filters, stride \1
Depth Concatenation
BN, ReLu
1x1x1 conv, 192 filters, stride \1
2x2x2 Avg pool, stride \2
BN, ReLu
1x1x1 conv, 192 filters, stride \1
BN, ReLu
3x3x3 conv, 32 filters, stride \1
Depth Concatenation
BN, ReLu
1x1x1 conv, 192 filters, stride \1
BN, ReLu
3x3x3 conv, 32 filters, stride \1
Depth Concatenation
BN, ReLu
1x1x1 conv, 256 filters, stride \1
BN, ReLu
1x1x1 conv, 256 filters, stride \2
2x2x2 Avg pool, stride \2
BN, ReLu
1³
3D- Global Average Pooling (GAP)
256
Fully Connected (FC)
Softmax

Fig. 4 Architecture of 3D-StudentNet

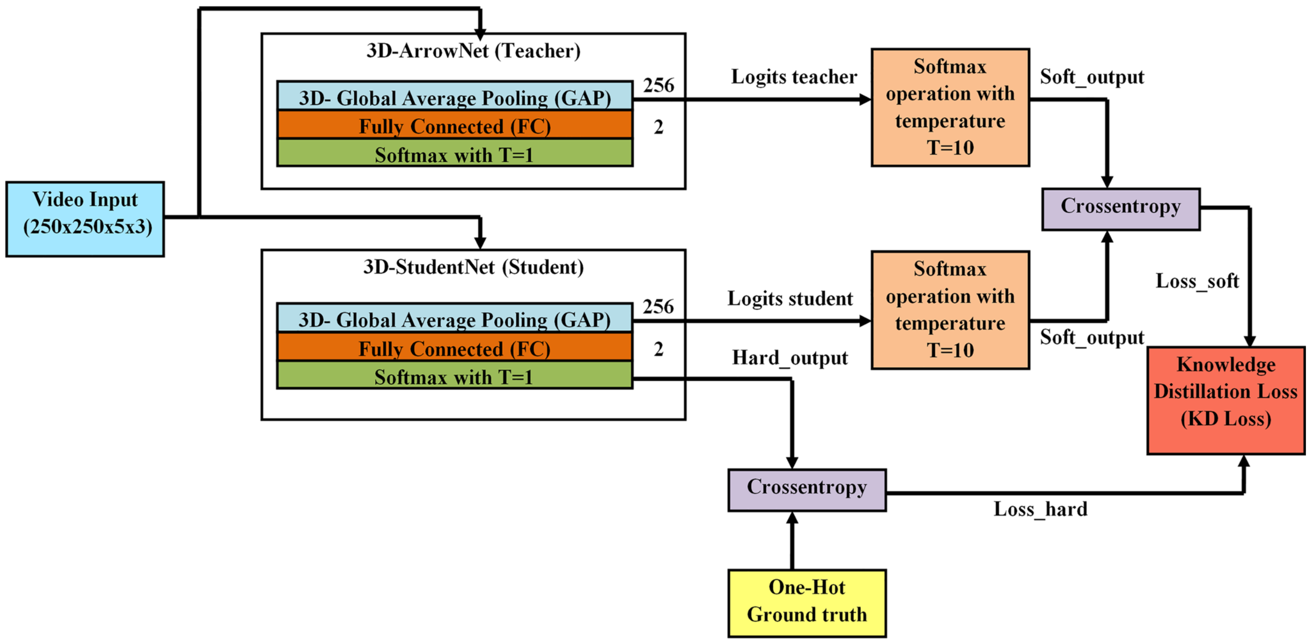


Fig. 5 Pipeline of knowledge distillation

to the frequency with which False Acceptance errors occur. Thus, False Acceptance rate = (Number of FA/Number of Attacks) * 100. False Rejection (FR) refers to a situation in which a presented video is “Real”, but the system classified it as an “Attack”. False Rejection rate (FRR) refers to the frequency with which False Rejection errors occur. Thus, False Rejection rate = (Number of FR/Number of Real attempts) * 100. The Average Classification Error Rate (ACER) is calculated to evaluate the FRADS performance. The ACER is the average of FAR and FRR. A lesser value of ACER indicates superior performance. ACER can be obtained using the confusion matrix, as shown in Eq. (4).

$$\text{ACER} = \frac{\left(\frac{\text{FP}}{\text{TN+FP}} + \frac{\text{FN}}{\text{FN+TP}}\right)}{2} \times 100 \tag{4}$$

For ideal performance, the accuracy must be 100%, and the FAR, FRR, and ACER (or half total error rate: HTER) values must equal zero. Table 5 provides the Accuracy, FAR, FRR, and ACER obtained on three datasets: RA, RM and combined dataset for the models M1 to M5. The RA database contains low-resolution captures and display devices that do not reflect recent mobile technologies. Spoof samples from RA are easier to detect than RM spoof samples because the RM database includes high-resolution recordings and high-quality displays. Thus, the performance obtained with the RA benchmark is superior to the RM benchmark. However, combined datasets are more challenging because they contain subjects, camera devices, display devices, and printer types from both datasets. There are also differences in recording environments between the two datasets. Consequently, achieving excellent performance on a combined dataset is more difficult than on a single dataset.

2D-VGG16 has the most trainable network parameters (134.2 Million) among all other backbones. It achieves 97.7, 97.3, and 91.3% accuracy on RA, RM, and combined datasets, respectively. Even though the RA benchmark FAR is zero, the FRR and ACER values are extremely high at 13.77 and 6.87. Additionally, it has high FAR, FRR, and ACER values for other benchmarks. Even though 2D-MobileNet-V2 has lesser computational complexity (0.3 G Flops, 2.2 Million trainable network parameters) than other existing backbones, as shown in Table 3, its efficiency is poor compared to the other backbones’ performance. It obtains an accuracy of 95.8, 98 and 93.8% accuracy on RA, RM and combined datasets. Also, it has very high FAR, FRR, and ACER values. 2D-ResNet18 outperforms MobileNet-V2 and VGG16 on a combined dataset with 11.1 Million trainable network parameters, 96.5% accuracy, and an ACER of 5.67. For all the datasets, 2D-DenseNet201 has superior performance than 2D-ResNet18. The performance of 3D-ArrowNet (with an accuracy of 100% and 99.66% for RA and RM datasets, respectively) is superior to 2D-DenseNet201 (with an accuracy of 99.7% and 98.01% for the RA and RM datasets, respectively). For the combined dataset, the performance of 2D-DenseNet201 (with an accuracy of 99.3%) and 3D-ArrowNet (with an accuracy of 99.23%) is nearly identical. 2D-DenseNet201 has 18.1 million trainable network parameters and requires 4G Flops. 3D-ArrowNet has only 1.3 million trainable network parameters but requires 12G Flops because 2D-DenseNet201 captures only spatial features, whereas 3D-ArrowNet captures both spatial and temporal features. Generally, false acceptance is typically riskier in biometrics than false rejection. Additionally, the face replay attack system should not delay or fail to authenticate the legitimate user. Therefore, FAR and FRR should ideally be zero. Table 6 depicts the experimental outcomes of 3D-studentNet on the RA, RM, and combined datasets before and after knowledge distillation (KD). 3D-StudentNet (After KD), with an accuracy of 100%, 99.66%, and 98.98% for the RA, RM, and combined datasets, respectively, outperforms 3D-StudentNet (Before KD), which achieved an accuracy of 97.08%, 98.67%, and 98.85% for the RA, RM, and combined datasets, respectively. 3D-StudentNet (After KD) obtains lower FAR, FRR, and ACER values than 3D-StudentNet (Before KD).

5 Experimental discussion

This section compares the results of the proposed 3D-FRADS models with state-of-the-art techniques on Replay Attack and Replay Mobile datasets. Tables 7 and 8 provide a comparison against several state-of-the-art techniques applied to RA and RM datasets, respectively. Most of these works have employed ACER/HTER as a performance metric.

The following frameworks are evaluated on the Replay-Attack benchmark. The efficiency of FRADS was evaluated on different facial features such as full-face, cropped face, eyes, nose, and mouth. To extract Multi-block LBP features, the image was divided into multiple blocks, LBP was applied to each block, and their histograms were concatenated to form the feature vector. Later, PCA was utilized to reduce the dimensionality of the feature vector. Subsequently, the reduced feature was fed into the SVM classifier. However, this method resulted in a low-performance HTER of 6.98 [50]. Transfer learning of the VGGFace model was employed to extract spatial features followed by LSTM with attention module-captured temporal information. EMM was utilized for motion magnification in video frames. This method obtained an HTER of 3.53 [41]. The shape-from-shading algorithm calculated albedo, reflectance, and depth for each RGB channel, concatenated them into nine channels, and input them into a shallow CNN classifier. This algorithm achieved an HTER of 3.1 [51]. A two-stream modified ResNet18-based network extracted features from a face image's High-frequency and Low-frequency components. Cross-frequency spatial and self-channel attentions were introduced to prioritize output feature maps requiring attention. Three steganalysis-rich-model (SRM) high-pass filters and three Gaussian blur filters extracted the high-frequency and low-frequency components, respectively. It achieved an

Table 7 Comparison with state-of-the-art techniques on replay attack dataset

Replay-Attack			
Year	References	Method	ACER/HTER (%)
June 2019	[43]	Two stream RGB and multi-scale retinex (MSR) with modified ResNet18 backbone	0.389
Feb 2020	[53]	CNN with perturbation convolutional layer	0.3
March 2020	[41]	Eulerian motion magnification in video frames + VGGFace + LSTM	3.53
April 2020	[51]	Shape-from-shading (SFS) algorithm + Shallow CNN network	3.1
Aug 2020	[44]	Fusing High and Low Frequency Features	3.1
Sep 2020	[52]	Siamese network with SqueezeNet backbone (triplet loss)	1.5
Sep 2020	[45]	Deep Reinforcement Learning	0
Oct 2020	[42]	Siamese network with VGG19 backbone (triplet loss)	0.7
March 2023	[50]	Multi-block LBP features	6.98
June 2023	[48]	Diffusion + (MobileNet with sigmoid)	0.09
June 2023	[46]	Siamese network with ResNet50V2 backbone	0.0375
Proposed	**M5**	**3D-ArrowNet**	**0**
Proposed	**M6**	**3D-StudentNet (After KD)**	**0**

Bold values indicate the results of the proposed work

Table 8 Comparison with state-of-the-art techniques on Replay Mobile dataset

Replay-Mobile				
Year	References	Method	Accuracy (%)	ACER/HTER (%)
Jan 2019	[54]	LTSS features from rPPG + SVM classifier	Not reported	32.5
Nov 2019	[40]	Hybrid (LBP + VGG16)	90.5	Not reported
Sep 2021	[47]	MTCNN + Scrambled face patches + denseNet-161	100	0
June 2023	[48]	Diffusion to improve class discrimination features	99.04	1.14
June 2023	[46]	Siamese network with ResNet50V2 backbone	100	0
Proposed	**M5**	**3D-ArrowNet**	**99.7**	**0.45**
Proposed	**M6**	**3D-StudentNet (After KD)**	**99.7**	**0.45**

Bold values indicate the results of the proposed work

HTER of 3.1 [44]. A Siamese network with triplet loss based on the SqueezeNet v1.1 architecture was implemented for Android OS. The custom data set RECOD-MPAD was employed. It obtained an HTER of 1.5 [52]. The research employed a Siamese network with a VGG19 backbone. The model was trained using the triplet loss function to reduce intra-class distance and maximize inter-class distance. The system achieved an HTER of 0.7 [42]. A research study utilized a multi-scale retinex model to separate an image's light and reflectance components. The reflectance component was then used to detect attacks. The study employed two input streams: the RGB input and the multi-scale retinex. The two streams were combined using an attention-based fusion technique. The backbone architecture used in the study was a modified ResNet18 model. The method achieved a 0.39 HTER [43]. The study employed a perturbation convolutional layer to combine the features of the first convolutional layer with LBP. The perturbation feature maps were obtained by taking the Hadamard product of the LBP feature with the first convolutional layer features. These perturbation feature maps were then used for classification, which resulted in an HTER of 0.3 [53]. The diffusion preprocessing approach was used to compress the images using inter-area interpolation and then enlarge them using pixel-area relationships to create diffused images. This diffusion scheme was more effective than the Perona-Malik and Gaussian filtering diffusion techniques. The image classification task employed MobileNet transfer learning and achieved an HTER of 0.09 [48]. In an interesting study, the preprocessing step involved resizing 25 frames from each video recording to 64 × 64. The resized frames were then enlarged to 224 × 224 using inter-area interpolation. The study used a Siamese network with a ResNet50V2 embedding network for classification. Instead of evaluating a single frame during testing, the model was fed with individual video frames. The predictions of each frame were averaged to classify the video. The study achieved a low HTER of 0.037. However, it's worth noting that the study used images instead of videos, which resulted in more training, validation, and testing samples than the proposed investigations. Additionally, the model used 2D-ResNet50v2, which only extracts spatial features and ignores temporal features. Also, the model was computationally intensive [46]. ResNet18 was modified to derive feature maps. This full feature map was fed to branch 1 to extract global features, while branch 2, with Gated recurrent unit (GRU) and linear layers, received the sub-patch feature map to extract local features. Finally, the two features were fused for classification. Reinforcement learning was used to locate sub-patches. This framework achieved an HTER of 0 but required a modified 2DResNet18 backbone and branch 1 network with 16.5 M trainable parameters. Branch 2 had dense layers and GRUs, which were computationally intensive. GRUs were used to leverage sub-patch features, not to extract temporal information [45]. As shown in Table 7, the proposed dynamic FRADS models, 3D-ArrowNet and 3D-StudentNet, achieve state-of-the-art performance on the Replay-Attack benchmark, obtaining 0 ACER.

The following frameworks were evaluated on the Replay-Attack benchmark. Remote Photoplethysmography (rPPG) is a method used to determine the pulse rate by analyzing the green color fluctuations on skin pixels caused by blood flow in skin-recorded video frames. Various algorithms like Li CVPR, the CHROM approach, and the Spatial Subspace Rotation (SSR) were used to get pulse signals. These pulse signals were then classified using binary classification algorithms such as SVM, Multi-Layer Perceptron (MLP), and Linear Discriminant Analysis (LDA). To get an accurate reading, it was important that the subject remained still during capture and ensure adequate lighting. Factors like makeup, compression techniques, video capture quality, lighting variations, and the subject's distance from the camera all affect the accuracy of the pulse signal extraction. The work achieved a low Half Total Error Rate (HTER) of 32.5 [54]. LBP features were most effective in detecting Low-image-quality attack frames, while VGG16 features were most effective in detecting high-image-quality attack frames. Combined features were utilized to achieve better classification results, leading to an accuracy of 90.52% [40]. The MobileNet with Diffusion technique was used to improve class discrimination features. It obtained an HTER of 1.14 [48]. An interesting work utilized multi-task cascaded convolutional networks to detect and align faces. The method involved scrambling and recombining 7 × 7 face patches to generate a new image of the scrambled face. The image's pixels were labeled '1' for real patches and '0' for attack patches. The DenseNet-161 model produced a 14 × 14 × 1 pixel-wise label feature map and the training process for the entire framework involved pixel-wise binary cross-entropy loss. The model achieved zero HTER. However, the method was computationally intensive due to the preprocessing steps it employed, such as shuffling and stitching sub-patches and the 2D-DenseNet-161 architecture. Also, 2D-DenseNet-161 does not consider temporal features [47]. A Siamese network with ResNet50V2 backbone networks was used for classification. Unlike traditional methods that test only a single frame during testing, this model was provided with all individual video frames for testing. To classify the video, the predictions made for each frame were averaged. The work achieved a low HTER of 0. However, the work used images instead of videos, which required more training, validation, and testing samples than the proposed investigations. Additionally, the model used 2D-ResNet50v2 to extract spatial features without considering temporal features. Furthermore, 2D-ResNet50v2 was computationally intensive [46].

As shown in Table 8, the proposed dynamic FRADS models, 3D-ArrowNet and 3D-StudentNet, achieve state-of-the-art performance on the Replay-Mobile benchmark, obtaining 0.45 ACER. The 100% accurate classification results obtained by [46, 47] are comparable to the 99.67% accuracy achieved by the proposed 3D-StudentNet backbone (After KD) with fewer network parameters. In the studies conducted by [46, 47] the model was fed with individual video frames instead of evaluating a single frame during testing. The predictions of each frame were averaged to classify the video. As a result, the studies achieved a low HTER of 0. In the proposed work, good accuracy can be achieved if scoring is done by averaging all the five-frame level scores in a video. The proposed 3D-StudentNet architecture considers both spatial and temporal features. Overall, the dynamic FRADS 3D-StudentNet model has shown better accuracy, FAR, FRR, and ACER on the Replay Attack and Replay Mobile datasets as compared to the existing static FRADS backbone architectures (M1-M4) after leveraging knowledge distillation while having fewer network parameters. The proposed technique has resulted in an Average Classification Error Rate of 0 for the Replay-Attack dataset and 0.45 for the Replay-Mobile dataset.

6 Conclusion

The proposed research will aid in detecting replay attacks in face and lip-reading biometric systems. The models are designed to tackle the problem of face replay attacks from digital displays and high-quality printed photos. This paper introduces two dynamic FRADS models: 3D-ArrowNet and 3D-StudentNet. These models only use RGB input and do not require additional expensive depth, infrared, or thermal sensors. Considering a small background area, the face ROI is obtained from five frames of an RGB video. The extracted ROI is then converted to the HSV color space and input to the proposed dynamic FRADS models to derive spatial and temporal attack patterns. To increase the number of training samples and to redirect the models to focus on attack patterns rather than face structure, the rotation of video frames is implemented as a data augmentation technique. The proposed 3D-ArrowNet architecture significantly outperforms existing static FRADS architectures, including VGG16, ResNet18, MobileNet-V2, and DenseNet201, as demonstrated by experimental results on two challenging databases, Replay Attack and Replay Mobile, as well as their aggregated dataset. This improvement is achieved through depth-wise concatenating four streams, each with a different filter size. 3D-StudentNet is created to reduce the computational complexity of 3D-ArrowNet. However, this leads to a decrease in the performance of 3D-StudentNet. To address this issue, knowledge distillation is used, where a trained 3D-ArrowNet is used as a teacher to train a 3D-StudentNet model. This significantly improves the performance of 3D-StudentNet to match its teacher's accuracy.

Future work can come up with more inventive 3D architecture with fewer network parameters. Training and testing on combined datasets improves model generalization. In addition, the model is capable of being trained for new types of attacks, such as 3D-mask attacks, cut photo and wrapped photo attacks, and partial face region covering attacks. Training of the 3D-StudentNet backbone with contrastive loss (Siamese network) or Arcface loss can further improve FRADS performance.

Acknowledgments The authors are grateful to PES University, Bangalore, for supporting the ongoing research work. We express our gratitude to the Reviewers for their valuable suggestions to enhance the quality and readability of the paper.

Author contributions Conceptualization: [Preethi S.J, Niranjana Krupa B]; Methodology: [Preethi S.J]; Software: [Preethi S.J]; Formal analysis and investigation: [Niranjana Krupa B]; Visualization: [Preethi S.J, Niranjana Krupa B], Validation: [Preethi S.J, Niranjana Krupa B], Writing—original draft preparation: [Preethi S.J]; Writing—review and editing: [Preethi S.J, Niranjan Krupa B]; Supervision: [Niranjana Krupa B].

Funding No funding was received for conducting this study.

Data availability We adopt Replay-Attack and Replay-Mobile datasets available for research. Code will be available on request. The Replay-Attack and Replay-Mobile datasets utilized in the current study are available at:
https://www.idiap.ch/en/scientific-research/data/replayattack
https://www.idiap.ch/en/scientific-research/data/replay-mobile

Declarations

Conflict of interest The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Ethical approval Authors are responsible for correctness of the statements provided in the manuscript.

References

1. Wang, C.: A REVIEW on 3D convolutional neural network. In: 2023 IEEE 3rd International Conference on Power, Electronics and Computer Applications (ICPECA). IEEE, pp. 1204–1208 (2023)
2. Seegehalli, P.J., Krupa, B.N.: Deep hybrid architectures and DenseNet35 in speaker-dependent visual speech recognition. SIViP (2024). https://doi.org/10.1007/s11760-024-03123-2
3. Li, L., Feng, X., Boulkenafet, Z., Xia, Z., Li, M., Hadid, A.: An original face anti-spoofing approach using partial convolutional neural network. In: 2016 Sixth International Conference on Image Processing Theory, Tools and Applications (IPTA). pp. 1–6 (2016)
4. Song, X., Zhao, X., Fang, L., Lin, T.: Discriminative representation combinations for accurate face spoofing detection. Pattern Recogn. **85**, 220–231 (2019)
5. Li, L., Feng, X.: Face anti-spoofing via deep local binary pattern. In: Jiang, X., Hadid, A., Pang, Y., Granger, E., Feng, X. (eds.) Deep learning in object detection and recognition, Springer, Singapore, pp. 91–111 (2019)
6. Khammari, M.: Robust face anti-spoofing using CNN with LBP and WLD. IET Image Proc. **13**, 1880–1884 (2019)
7. Chen, H., Chen, Y., Tian, X., Jiang, R.: A cascade face spoofing detector based on face anti-spoofing R-CNN and improved retinex LBP. IEEE Access. **7**, 170116–170133 (2019)
8. Feng, L., Po, L.-M., Li, Y., Xu, X., Yuan, F., Cheung, T.C.-H., Cheung, K.-W.: Integration of image quality and motion cues for face anti-spoofing: a neural network approach. J. Vis. Commun. Image Represent. **38**, 451–460 (2016)
9. Asim, M., Ming, Z., Javed, M.Y.: CNN based spatio-temporal feature extraction for face anti-spoofing. In: 2017 2nd International Conference on Image, Vision and Computing (ICIVC). pp. 234–238 (2017)
10. Li, L., Xia, Z., Hadid, A., Jiang, X., Zhang, H., Feng, X.: Replayed video attack detection based on motion blur analysis. IEEE Trans. Inf. Forensics Secur. **14**, 2246–2261 (2019)
11. Menotti, D., Chiachia, G., Pinto, A., Schwartz, W.R., Pedrini, H., Falcão, A.X., Rocha, A.: Deep representations for iris, face, and fingerprint spoofing detection. IEEE Trans. Inf. Forensics Secur. **10**, 864–879 (2015)
12. Lucena, O., Junior, A., Moia, V., Souza, R., Valle, E., Lotufo, R.: Transfer learning using convolutional neural networks for face anti-spoofing. In: Image Analysis and Recognition: 14th International Conference, ICIAR 2017, Proceedings 14. Springer International Publishing, Montreal, QC, Canada, pp. 27–34, (2017)
13. Ur Rehman, Y.A., Po, L.M., Liu, M.: Deep learning for face anti-spoofing: an end-to-end approach. In: 2017 Signal Processing: Algorithms, Architectures, Arrangements, and Applications (SPA). IEEE, pp. 195–200 (2017)
14. Li, L., Xia, Z., Li, L., Jiang, X., Feng, X., Roli, F.: Face anti-spoofing via hybrid convolutional neural network. In: 2017 International Conference on the Frontiers and Advances in Data Science (FADS). IEEE, pp. 120–124 (2017)
15. Atoum, Y., Liu, Y., Jourabloo, A., Liu, X.: Face anti-spoofing using patch and depth-based CNNs. In: 2017 IEEE International Joint Conference on Biometrics (IJCB). IEEE, pp. 319–328 (2017)
16. Luo, S., Kan, M., Wu, S., Chen, X., Shan, S.: Face anti-spoofing with multi-scale information. In: 2018 24th International Conference on Pattern Recognition (ICPR). IEEE, pp. 3402–3407 (2018)
17. Botelho de Souza, G., Papa, J.P., Marana, A.N.: On the learning of deep local features for robust face spoofing detection. In: 2018 31st SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI). IEEE, pp. 258–265 (2018)
18. Rehman, Y.A.U., Po, L.M., Liu, M.: LiveNet: improving features generalization for face liveness detection using convolution neural networks. Expert Syst. Appl. **108**, 159–169 (2018)
19. Larbi, K., Ouarda, W., Drira, H., Ben Amor, B., Ben Amar, C.: DeepColorFASD: face anti spoofing solution using a multi channeled color spaces CNN. In: 2018 IEEE International Conference on Systems, Man, and Cybernetics (SMC). IEEE, pp. 4011–4016 (2018)
20. Rehman, Y.A.U., Po, L.-M., Liu, M., Zou, Z., Ou, W., Zhao, Y.: Face liveness detection using convolutional-features fusion of real and deep network generated face images. J. Vis. Commun. Image Represent. **59**, 574–582 (2019)
21. Hao, H., Pei, M., Zhao, M.: Face liveness detection based on client identity using siamese network. In: Pattern Recognition and Computer Vision: Second Chinese Conference, PRCV 2019, Xi'an, China. Springer International Publishing, pp. 172–180 (2019)
22. Deb, D., Jain, A.K.: Look locally infer globally: a generalizable face anti-spoofing approach. IEEE Trans. Inf. Forensics Secur. **16**, 1143–1157 (2020)
23. Chen, B., Yang, W., Li, H., Wang, S., Kwong, S.: Camera invariant feature learning for generalized face anti-spoofing. IEEE Trans. Inf. Forensics Secur. **16**, 2477–2492 (2021)
24. Abdullakutty, F., Elyan, E., Johnston, P., Ali-Gombe, A.: Deep transfer learning on the aggregated dataset for face presentation attack detection. Cogn. Comput. **14**, 2223–2233 (2022)
25. Liu, A., Tan, Z., Liang, Y., Wan, J.: Attack-agnostic deep face anti-spoofing. In: Presented at the Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (2023)
26. Zeng, D., Gao, L., Fang, H., Xiang, G., Feng, Y., Lu, Q.: Bandpass filter based dual-stream network for face anti-spoofing. In: 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW). pp. 6403–6410 (2023)
27. Xu, Z., Li, S., Deng, W.: Learning temporal features using LSTM-CNN architecture for face anti-spoofing. In: 2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR). IEEE, pp. 141–145 (2015)
28. Tu, X., Fang, Y.: Ultra-deep neural network for face anti-spoofing. In: In Neural Information Processing: 24th International Conference, ICONIP 2017, Guangzhou, China, Proceedings, Part II 24. Springer International Publishing pp. 686–695 (2017)
29. Lin, C., Liao, Z., Zhou, P., Hu, J., Ni, B.: Live face verification with multiple instantialized local homographic parameterization. In: Proceedings of the 27th International Joint Conference on Artificial Intelligence, Stockholm, Sweden, pp. 814–820 (2018)
30. Yang, X., Luo, W., Bao, L., Gao, Y., Gong, D., Zheng, S., Li, Z., Liu, W.: Face anti-spoofing: model matters, so does data. In: 2019

IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, Long Beach, CA, USA, pp. 3502–3511 (2019)
31. Muhammad, U., Usman, M., Holmberg, T.: Face anti-spoofing via sample learning based recurrent neural network (RNN). In: The British Machine Vision Conference (2019)
32. Ming, Z., Yu, Z., Al-Ghadi, M., Visani, M., Luqman, M.M., Burie, J.-C.: Vitranspad: video transformer using convolution and self-attention for face presentation attack detection. In: 2022 IEEE International Conference on Image Processing (ICIP). IEEE, pp. 4248–4252 (2022)
33. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition, (2015)
34. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
35. Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 2261–2269 (2017)
36. Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L.-C.: MobileNetV2: inverted residuals and linear bottlenecks. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4510–4520 (2018)
37. Chingovska, I., Anjos, A., Marcel, S.: On the effectiveness of local binary patterns in face anti-spoofing. In: 2012 BIOSIG - Proceedings of the International Conference of Biometrics Special Interest Group (BIOSIG). IEEE, pp. 1–7 (2012)
38. Costa-Pazo, A., Bhattacharjee, S., Vazquez-Fernandez, E., Marcel, S.: The replay-mobile face presentation-attack database. In: 2016 International Conference of the Biometrics Special Interest Group (BIOSIG). IEEE, pp. 1–7 (2016)
39. King, D.E.: Dlib-ml: A machine learning toolkit. J. Mach. Learn. Res. **10**, 1755–1758 (2009)
40. Das, P.K., Hu, B., Liu, C., Cui, K., Ranjan, P., Xiong, G.: A new approach for face anti-spoofing using handcrafted and deep network features. In: 2019 IEEE International Conference on Service Operations and Logistics, and Informatics (SOLI). IEEE, pp. 33–38 (2019)
41. Ge, H., Tu, X., Ai, W., Luo, Y., Ma, Z., Xie, M.: Face anti-spoofing by the enhancement of temporal motion. In: 2020 2nd International Conference on Advances in Computer Technology, Information Science and Communications (CTISC). IEEE, pp. 106–111 (2020)
42. Li, L., Xia, Z., Jiang, X., Roli, F., Feng, X.: CompactNet: learning a compact space for face presentation attack detection. Neurocomputing **409**, 191–207 (2020)
43. Chen, H., Hu, G., Lei, Z., Chen, Y., Robertson, N.M., Li, S.Z.: Attention-based two-stream convolutional networks for face spoofing detection. IEEE Trans. Inf. Forensics Secur. **15**, 578–593 (2019)
44. Chen, B., Yang, W., Wang, S.: Face anti-spoofing by fusing high and low frequency features for advanced generalization capability. In: 2020 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR). IEEE, pp. 199–204 (2020)
45. Cai, R., Li, H., Wang, S., Chen, C., Kot, A.C.: DRL-FAS: a novel framework based on deep reinforcement learning for face anti-spoofing. IEEE Trans. Inf. Forensics Secur. **16**, 937–951 (2020)
46. Alassafi, M.O., Ibrahim, M.S., Naseem, I., AlGhamdi, R., Alotaibi, R., Kateb, F.A., Oqaibi, H.M., Alshdadi, A.A., Yusuf, S.A.: Fully supervised contrastive learning in latent space for face presentation attack detection. Appl. Intell. **53**, 21770–21787 (2023)
47. Kantarcı, A., Dertli, H., Ekenel, H.K.: Shuffled patch-wise supervision for presentation attack detection. In: 2021 International Conference of the Biometrics Special Interest Group (BIOSIG). IEEE, pp. 1–5 (2021)
48. Alassafi, M.O., Ibrahim, M.S., Naseem, I., AlGhamdi, R., Alotaibi, R., Kateb, F.A., Oqaibi, H.M., Alshdadi, A.A., Yusuf, S.A.: A novel deep learning architecture with image diffusion for robust face presentation attack detection. IEEE Access. **11**, 59204–59216 (2023)
49. Hinton, G., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network, (2015)
50. Günay Yılmaz, A., Turhal, U., Nabiyev, V.: Face presentation attack detection performances of facial regions with multi-block LBP features. Multimed. Tools Appl. **82**, 40039–40063 (2023)
51. Pinto, A., Goldenstein, S., Ferreira, A., Carvalho, T., Pedrini, H., Rocha, A.: Leveraging shape, reflectance and albedo from shading for face presentation attack detection. IEEE Trans. Inf. Forensics Secur. **15**, 3347–3358 (2020)
52. Almeida, W.R., Andaló, F.A., Padilha, R., Bertocco, G., Dias, W., Torres, R. da S., Wainer, J., Rocha, A.: Detecting face presentation attacks in mobile devices with a patch-based CNN and a sensor-aware loss function. PLOS ONE. 15, e0238058 (2020)
53. Rehman, Y.A.U., Po, L.-M., Komulainen, J.: Enhancing deep discriminative feature maps via perturbation for face presentation attack detection. Image Vis. Comput. **94**, 103858 (2020)
54. Heusch, G., Marcel, S.: Remote blood pulse analysis for face presentation attack detection. In: Handbook of Biometric Anti-Spoofing: Presentation Attack Detection. Springer International Publishing, pp. 267–289 (2019)