CNN-LPQ: convolutional neural network combined to local phase quantization based approach for face anti-spoofing

Mebrouka Madi, Mohammed Khammari, Mohamed-Chaker Larabi

CNN-LPQ: Convolutional Neural Network combined to Local Phase Quantization based approach for face anti-spoofing

Mebrouka Madi[1], Mohammed Khammari[1], Mohamed-Chaker Larabi[2]

[1]Laboratory of Medical Informatics (LIMED), Faculty of Exact Sciences, University of Bejaia, 06000 Bejaia, Algeria.
[2]CNRS, Université de Poitiers, XLIM, UMR 7252, 86073 Poitiers, France.

Contributing authors: mebrouka.madi@univ-bejaia.dz; mohammed.khammari@univ-bejaia.dz; chaker.larabi@univ-poitiers.fr;

Abstract

In this paper, we propose a novel approach for face spoofing detection using a combination of color texture descriptors with a new convolutional neural network (CNN) architecture. The proposed approach is based on a new convolutional neural network architecture composed of two CNN parallel branches. The first branch is fed with complementary shallow local phase quantization (LPQ) invariant descriptors that result from joint color texture information from the hue, saturation, and value (HSV) color space to accurately capture the reflection properties of the face. Combining the HSV color space with LPQ is known to significantly improve performance. The second branch of the CNN takes an RGB image directly as input, effectively separating chromatic (color-related) information from achromatic (brightness-related) information in order to extract crucial facial color features. Each branch of the CNN produces a vector of deep features that are extracted. To effectively concatenate the deep features from the two output branches, we employ an attention mechanism based combination method. This method captures the complementarity of the two branches, improving the accuracy and robustness of the model. The combined feature vectors form an input vector for the next Dense layer, where the model can distinguish between live and spoofed faces. Our method detects 2D facial spoofing attacks involving printed photos and replayed videos. We showcase the effectiveness and superior performance of our approach through a series of experiments conducted on both the CASIA-FASD and Replay-Attack datasets. Our results are promising and

surpassing those of other state-of-the-art methods on both used datasets in terms of 9 performance metrics.

Keywords: Deep learning . Convolutional neural network . Images classification . Swish activation function . Artificial neural network . RGB color space . HSV color space . Local phase quantization . Attention mechanism . Computer vision . Artificial intelligence . Shallow feature . Deep feature

1 Introduction

Facial recognition is a biometric system that utilizes facial images to verify and authenticate individuals. In the context of increasing security concerns, face authentication plays a crucial role in ensuring the integrity and reliability of identification processes. However, face spoofing attacks pose a significant threat to the accuracy and security of facial recognition systems. These attacks involve the use of counterfeit photos or videos to deceive the system. To counter such threats, the development of effective face anti-spoofing techniques is essential. These techniques aim to detect and prevent fraudulent attempts by distinguishing between genuine facial images and spoofed ones. Implementing robust face anti-spoofing measures alongside facial recognition is crucial to ensure secure and reliable authentication processes.

In recent years, numerous face anti-spoofing methods have been proposed to address the vulnerability of existing face recognition systems to various spoofing attacks [2, 3]. Studies have shown that these systems are susceptible to different types of face spoofing attacks [4]. These attacks can be categorized into three main types: print attacks, replay attacks, and 3D mask attacks. Print attacks involve the use of printed photos or images to deceive the face biometric systems. Replay attacks exploit face videos to gain unauthorized access. These two types of attacks fall under the category of 2D face attacks. On the other hand, 3D mask attacks utilize realistic soft plastic masks resembling human skin, making it more challenging to develop accurate countermeasures against them. Addressing these multiple face spoofing attacks remains a significant research challenge in the field.

Various approaches were introduced in the last decade to address face spoofing attacks. They can be categorized into three groups: 1) Texture-based methods, which exploit the differences in texture characteristics between genuine face images and those generated by different types of attacks. These techniques are faster as they can operate with just one image sample instead of a complete video sequence; 2) Motion-based methods, which are effective in detecting static presentation attacks like photo attacks. However, they may not be as effective against video replay attacks that display liveness information such as eye blinking and head movements. For example, Anjos et al. [1] proposed a motion model that analyzes the relationship between the face region and the background to distinguish real faces from photographic faces; and 3) Image quality-based methods, which analyze the image distortion typically present in spoofed face images. Several techniques rely on image quality analysis, including deformation, frequency, impairment, and color. Frequency domain analysis, such as the one utilized

in Zhang et al. [5], considers high-frequency information as an indicator of image quality. Hence image quality features using multiple difference of gaussian (DoG) filters are exploited to detect fake faces. However, frequency-based approaches may fail when high-quality spoof images or videos are presented to the camera.

These different approaches offer valuable insights into face anti-spoofing methods, each with its own strengths and limitations. Nonetheless, further research is needed to develop more robust and accurate techniques to address the challenges posed by face spoofing attacks. This paper focuses on both print and replay attacks in the context of face anti-spoofing detection.

In summary, the main contributions of our paper are as follows:

1. We propose a novel approach for face spoofing detection, involving printed photos and replayed videos, using a combination of color texture descriptors with a new convolutional neural network (CNN) architecture.
2. Our approach is based on a new convolutional neural network architecture composed of two CNN parallel branches:
   (a) The first branch is fed with complementary shallow local phase quantization (LPQ) invariant descriptor that results from joint color-texture information from the hue, saturation, value (HSV) color space to accurately capture the reflection properties of the face. Combining the HSV color space with LPQ is known to significantly improve performance.
   (b) The second branch of the CNN takes an RGB image directly as input, effectively separating chromatic (color-related) information from achromatic (brightness-related) information to extract crucial facial color features.
3. Each branch of the CNN produces a vector of deep features that are extracted. To effectively concatenate the deep features from the two output branches, we employ an attention mechanism based combination method. This method captures the complementarity of the two branches, improving the accuracy and robustness of the model. The combined feature vectors form an input vector for the next Dense layer, where the model can distinguish between live and spoofed faces.
4. We conducted a series of experiments on both CASIA-FASD and Replay-Attack datasets to demonstrate the effectiveness and the superior performance of our approach.
5. Our results are promising and outperform those of other state-of-the-art methods.

The remainder of the paper is organized as follows: In Section 2, we briefly present a review of the state-of-the-art face anti-spoofing methods. We describe the proposed approach in Section 3. The performance evaluation and results are presented in Section 4. Finally, Section 5 concludes the paper by identifying promising directions for future work.

2 Related work

In this section, we provide a comprehensive review of state-of-the-art face anti-spoofing methods, categorizing them into three groups: hand-created features-based methods, CNN-based methods, and a combination of hand-crafted features with CNN-based methods.

2.1 Handcrafted features-based methods

Traditional face anti-spoofing techniques typically involve extracting handcrafted features from facial images. These features are then utilized in conjunction with classification algorithms such as support vector machine (SVM) [6] or linear discriminant analysis (LDA) [7] to develop anti-spoofing systems.

Määttä et al. [8] were the first to propose the application of multiple uniform LBP operators at three different scales: $LBP^{u2}_{8,2}$ (uniform circular LBP extraction applied to a neighborhood of 8 pixels with a radius of 2 pixels), $LBP^{u2}_{16,2}$ (uniform circular LBP extraction applied to a neighborhood of 16 pixels with a radius of 2 pixels), and $LBP^{u2}_{8,1}$ (uniform circular LBP extraction applied to a neighborhood of 8 pixels with a radius of 1 pixel). They then extracted texture feature histograms from local blocks of grayscale and global images, concatenated them to form a 531-dimensional feature histogram, and inputted it to an SVM classifier with RBF as the training kernel for face anti-spoofing. While the texture analysis algorithm relying on grayscale maps proves effective for high-quality images, such as those with high resolution, it encounters challenges in accurately distinguishing certain low-quality images. Additionally, Yang et al. [9] proposed the use of a face region referred to as the holistic face, which involves segmenting canonical facial regions, including the left eye region, right eye region, nose region, mouth region, and face contour. Low-level features are then extracted using various texture descriptors such as LBP [8], HOG [10], and LPQ [11]. Additionally, a high-level spatial pyramid descriptor [12] is created using the low-level features and a 512-word codebook. Finally, the histogram of these image representations is combined into a single feature vector and input into an SVM classifier to differentiate between real face presentations and Presentation Attacks (PAs). Similarly, Chingovska et al. [13] employed the LBP descriptor to extract texture features, calculating histograms in two different ways: either considering the entire image or dividing it into blocks and calculating histograms for each block independently. The concatenated histograms were then classified using LDA and SVM to distinguish between live and spoof images. They achieved an HTER of 18.2% on the CASIA-FASD dataset and 13.8% on the Replay-Attack dataset. For this reason, Boulkenaf et al. [14] proposed a method based on color texture analysis. This method extracts the characteristic LBP histograms of a single image channel in the three color spaces RGB, YCbCr and HSV, and concatenates them to form the final descriptor. This work has shown that the color texture-based method is superior to the gray texture-based method for detecting different attacks. Boulkenaf et al. [15] also proposed a robust method based on color texture and combined the multi-scale LBP features of the face in HSV space with LPQ features of the face in YCbCr space. Although the experiment yielded good results with EER of 0.40% and HTER of 2.80% on Replay-Attack, EER of 2.10% and EER of 4.90% on CASIA-FASD and MSU mobile, respectively, the low level of microtexture descriptors makes them susceptible to variations in lighting conditions and high-quality images. In order to improve discrimination further, in the same year, Boulkenaf et al. [16] proposed an advanced approach for anti-spoofing detection using Fisher Vector encoding on speeded-up robust features (SURF) features from different color spaces. The SURF descriptor was applied to each color band, and the resulting descriptors were concatenated to form a single feature vector called CSURF, and a

dimensionality reduction technique, principal component analysis (PCA), was applied to reduce the dimensionality of the concatenated feature vector. They achieved an EER of 0.10% and a HTER of 2.20% on Replay-Attack, EER of 2.80% and EER of 2.20% on CASIA-FASD and MSU mobile, respectively. This method showed excellent and more stable performance than previous methods. Wen et al. [17] proposed a face anti-spoofing detection method based on image distortion analysis (IDA). The four different features such as specular reflection, blur, color moment, and color diversity were extracted to form feature vectors IDA is fed into two SVM classifiers corresponding to photo attacks and video replay attacks respectively, and then a score-level fusion based on the min rule [19] is applied gives to distinguish real and fake faces. In contrast to texture feature-based methods, the generalization performance of this method showed promising. The quality of the image is highly dependent on the shooting equipment and the external conditions. External factors like bad lighting and low-quality camera equipment can also alter the appearance of a live human face. Singh et al. [18] proposed a method for liveliness detection based on eye blinking and mouth movement. The area of the eyes was done by checking for eye existence within the eye region and mouth movement is ascertained by checking for the existence of teeth within the mouth region using the HSV (hue, saturation, value) of the tooth were calculated to determine whether the eyes and the mouth were open. The subjects acted according to the phrase prompts randomly generated by the system and completed the relevant action to prove that it is a real face. Although the method had good accuracy in a few samples of various forms of attacks generated by the authors, the result also shows that the liveness test was bypassed by cut-photo attacks where both the eye and the mouth region are cut out.

2.2 CNN-based methods

In recent years, convolutional neural networks (CNNs) have demonstrated strong performance in the detection of presentation attacks. For frameworks based on different architectures, George et al. [20] proposed DeepPixBiS, a CNN based on DenseNet [21], which incorporates deep pixel-wise binary supervision for presentation attack detection. DeepPixBiS achieved a HTER of 0% on the Replay Mobile dataset and an ACER of 0.42% on protocol 1 of the Oulu-NPU dataset, thus surpassing state-of-the-art methods. It is crucial to leverage automatic generalization methods that rely on deep pixel-wise binary supervision to effectively combat spoofing attacks. Recently, Satapathy et al. [24] , Abdullakutty et al. [22], Abdullakutty et al. [23], Gwyn et al. [25] and Wang et al. [26] utilized various image classification models such as Inception-V2, ResNet-34, ResNet-18, ResNeXt-50, GoogleNet, VGG-19, AlexNet, ResNet-50, Inception-V3, VGG-16, DenseNet-121, Xception, MobileNetV2, and ShuffleNetV2 for spoofing attack detection. The result showed that Inception-V2, ResNet-34, ResNet-18, ResNeXt-50, GoogleNet, VGG-19, AlexNet, ResNet-50, Inception-V3, VGG-16, DenseNet-121, and Xception achieved scores, with accuracy of 94%, 92%, 92%, 91%, 88.00%, 83.00%, 83.00%, 93%, 86%, 85%, 93.00%, and 62.00%, respectively, tested on the CASIA-FASD dataset. MobileNetV2 obtained an EER of 9.40% (resp. 3.6%) and HTER of 16.7% (resp. 16.7%) on the CASIA-FASD and Replay-Attack datasets. Similarly, ShuffleNetV2 achieved an EER of 14.9% (resp. 6.33%) and HTER of 21.9%

(resp. 21.8%) on the CASIA-FASD and Replay-Attack datasets. Additional models focusing on different parts of CNN architecture are explored by various researchers. Li et al. [27] proposed utilizing a deep CNN based on the VGG-Face model to extract features from different layers of the CNN, combining them into a single feature for face anti-spoofing. Subsequently, block principal component analysis (PCA) is applied to reduce the feature dimension. The reduced feature is then input into an SVM for detecting photo and video replay attacks. This approach achieved an EER of 4.50% on the CASIA-FASD dataset and 2.9% on the Replay-Attack dataset, representing a significant improvement over previous state-of-the-art methods. The success can be attributed to the utilization of a deep CNN based on VGG-Face, which proves to be a potent tool for facial anti-spoofing. Yang et al. [28] proposed a novel spatio-temporal anti-spoofing network (STASN), which considers both local spatial and global temporal information to detect photo and video replay attacks. The STASN model consists of three components: the temporal anti-spoofing module (TASM), region attention module (RAM), and spatial anti-spoofing module (SASM). The TASM utilizes a CNN-LSTM architecture to process frame sequences as input, first extracts temporal features using CNN, then performs propagation using LSTM, and finally predicts the result of binary classification. The RAM learns CNN-based offsets produced by the TASM and identifies the regions associated with the sequence images. The SASM incorporates feature extraction from different selected local image regions (such as edges, moiré patterns, and reflected artifacts) obtained from the RAM output into a k-branch CNN with an attention mechanism for learning spatial texture features. STASN achieved an ACER of 1.0% on protocol 1 of the Oulu-NPU dataset and an ACER of 0.30% on protocol 1 of the SiW dataset. This approach demonstrates that integrating both spatio-temporal features can more comprehensively and efficiently distinguish between genuine and spoofing faces. Compared to methods that rely on a single feature, the integration of multiple features enhances accuracy while improving the robustness, generalizability, and performance of the face anti-spoofing algorithm model. Also, Deb et al. [29] introduced SSR-FCN, a self-supervised regional fully convolutional network that learns local discriminative cues for face anti-spoofing. This method achieved an ACER of 4.6% on protocol 1 of the Oulu-NPU dataset, making it effective at detecting spoofing attacks using deep supervision. In a different fashion, Shao et al. [30] employed a deep CNN with multi-adversarial learning and utilized learned generalized features to estimate depth maps and classify images, while Souza et al. [31] introduced a locally specialized CNN (LSCNN) model that focuses on deep local spoofing features. This method achieved an EER of 4.44% on the CASIA-FASD dataset, and an EER of 0.33%, and HTER of 2.50% on the Replay-Attack dataset. Furthermore, Sun et al. [32] proposed the DANet model with the DyAttention module, which incorporates a spatial attention mechanism for mask generation, and then applied dynamic activation to automatically detect and enhance clear texture features associated with spoof patterns in the facial area in a piecewise manner. DANet achieved an EER of 2.50%, an EER of 0.20%, and an ACER of 0.8% on the CASIA-FASD, Replay-Attack, and Oulu-NPU datasets, respectively. In addition, Kong et al. [33] proposed the SE-ResNet50 model with the residual convolution module, and which also incorporates a channel attention mechanism to extract different

feature expressions in the nose and cheek regions of the face. This method achieved an accuracy of 99.98% and 97.75% on the Replay-Attack and CASIA-FASD datasets, respectively. Subsequently, and for the combination purpose, Liu et al. [34] proposed to use a two-stream convolutional neural network (CNN) and recurrent neural network (RNN) architecture for face PAD. This approach comprises two parts: the CNN part estimates the depth map supervision to extract depth texture features to distinguish between real and fake faces, and the second RNN part learns to estimate the Photo-PlethysmoGraphy (rPPG) signal features, which checks the temporal variance of the video images. The face PAs are then identified without the use of a binary classifier by thresholding a score that is calculated using the weighted quadratic sum of the estimated depth map from the previous image of the video and the estimated rPPG signal features, and achieved an ACER of 1.6% on the Oulu-NPU dataset. Silva et al. [35] proposed a hybrid model that combines a residual spatial-temporal convolutional neural network (CNN) with a channel-separated CNN. This approach achieved improved performance in both live and spoof attack scenarios. They evaluated their model on the Oulu-NPU and SiW datasets. Furthermore, Xu et al. [36] first proposed an architecture based on long short-term memory (LSTM) and convolutional neural network (CNN) networks to learn the spatio-temporal features of an image for face anti-spoofing. The CNN is composed of several branches, and each branch is utilized to extract spatial texture features from an image, then LSTM units are connected at the end of each CNN branch to learn the temporal relations between images. Finally, all LSTM unit outputs are connected to a softmax as a binary classifier to differentiate genuine face presentation from APs attacks. They achieved an EER of 5.17% and an HTER of 5.93% on the CASIA-FASD dataset. The authors noted, as had other researchers before them, that adding extra background information to an image's initial recognized face via spatio-temporal analysis can aid face anti-spoofing methods. As a deep learning model tailored for 3D faces, Guo et al. [37] delved into high-fidelity face image synthesis using 3D face models and supervised deep network training. The obtained results include an ACER of 11.72% on the CASIA-FASD dataset, an EER of 2.22%, HTER of 1.67% on the CASIA-MFSD dataset, and an EER of 0.25%, HTER of 0.63% on the Replay-Attack dataset. The proposed model adeptly detects 3D mask attacks through comprehensive supervision, achieving state-of-the-art performance.

2.3 Hybrid methods

Due to the inherent difference of extraction, hand-crafted features, and CNN features provide a distinct way of characterizing the problem. To leverage the strengths of both features, researchers have focused on their fusion. Similar to the proposed method, several researchers have developed hybrid techniques for face spoofing detection. In [38], both hand-crafted and CNN features are directly concatenated, reduced, and utilized to train a classifier. For instance, Khammari [39] proposed a CNN based on CaffeNet [40] that combines local binary pattern (LBP) and simplified local weber descriptor (SWLD) features. The fusion is performed at the score level using a support vector machine (SVM) classifier to discriminate between genuine and fake faces. This method achieved an EER of 2.62%, and HTER of 2.14% on the CASIA-FASD dataset, and an EER of 0.53%, and HTER of 0.69% on the Replay-Attack dataset.

However, local descriptor-based features lose pixel-level details when compared to the original face input, which limits the performance of the model. On the other hand, dynamic features across temporal frames (such as motion, changes in lighting, and physiological signals) can also be useful inputs for CNNs. Similarly, Atoum et al. [41] present a two-stream CNN that combines patch-based texture cues and pseudo depth-map cues. The scores from both streams are weighted and summed to obtain the final score for distinguishing between real and planar face presentation attacks. This method achieved an EER of 2.67%, and HTER of 2.27% on the CASIA-FASD dataset, and an EER of 0.79 %, and HTER of 0.72% on the Replay-Attack dataset, an EER of 0.35 %, and HTER of 0.21% on the MSU-USSA dataset. Thus, a well-trained depth-map is capable of predict holistic depth maps as evidence of decision-making. Chen et al. [42] proposed a two-stream convolutional neural network (TSCNN) with an attention model for the fusion of features extracted from the RGB color space and the multi-scale retinex (MSR) space. This method achieved an EER of 3.14%, and an EER of 0.21 % on the CASIA-FASD and Replay-Attack datasets, respectively, and an HTER of 0.38 % on the CASIA-FASD. This approach shares similarities with our proposed work, particularly in the use of attention mechanisms to effectively combine information from different sources. In [43], Wang et al. combine learned deep texture features with representations derived from depth images. The fusion is performed at the score level to produce the final decision. They achieve an HTER of 2.3% on the CASIA-FASD dataset. Asim et al. [44] utilized a fusion of CNN features with LBP-TOP representing the spatial-temporal information. This method achieved an EER of 8.02%, and HTER of 9.94% on the CASIA-FASD dataset, and an EER of 3.22%, and HTER of 4.70% on the Replay-Attack dataset. The fact that the handcrafted features of this hybrid framework rely heavily on the well-trained convolutional features is one of its limitations. Additionally, it is still unclear whether the various types of handcrafted features are better suited for shallow or deep convolutional features. Antil et al. [45] proposed a two-stream framework that fuses multi-level elliptical local binary patterns (ELBP) texture features with modified Xception network-based deep features to learn highly discriminative features for face anti-spoofing. This achieved an EER of 2.37%, a HTER of 3.20% on the CASIA-FASD dataset. For the Replay-Attack dataset, it resulted in an EER of 0% and an HTER of 0%. On the MSU-USSA dataset, the method achieved an EER of 0% and an HTER of 0.06%. This approach shares similarities with our proposed work, particularly in the use of the Xception network. Feng et al. [46] introduced a CNN pre-trained model that fuses shearlet-based image quality features and optical flow-based motion features using multiple branches perceptron to detect anomalies in print attacks. In comparison to previous methods, the proposed method achieves a perfect EER of 0% and HTER of 0% on the Replay-Attack dataset, demonstrating its exceptional accuracy in detecting presentation attacks. While maintaining a near-perfect EER of 0% and HTER of 0.06% on the MSU-USSA dataset, further highlights the method's robustness across different datasets. However, head motions are easily replicated in a replay attack, rendering such dynamic cues less reliable. Additionally, Shu et al. [47] proposed a model called multi-scale color inversion dual-stream convolutional neural network (MSCI-DSCNN), consisting of two streams. One stream converts RGB images to grayscale and applies

multi-scale color inversion, while the other stream directly uses RGB images as input. The features extracted from both streams are combined and utilized for face spoofing detection. This method achieved an EER of 2.90% on the CASIA-FASD dataset, and an EER of 4.70%, and HTER of 0.39% on the Replay-Attack dataset, and an ACER of 1.6% on protocol 1 of the Oulu-NPU dataset.

Inspired by these works, we propose a novel approach for face spoofing detection based on the combination of handcrafted and CNN methods in different color spaces (HSV and RGB). Our proposed approach uses the LPQ descriptor to efficiently capture local structural texture information, and CNN based on a modified and reduced version of the Xception model to extract global and contextual features. This combination exploits the strengths of each method, improving the overall performance of our approach and enabling us to distinguish more accurately between live and spoofed (the different types of spoofing attacks) faces.

3 Proposed approach overview

In this section, we introduce our innovative method for detecting face spoofing through the integration of color texture descriptors with a novel CNN architecture. Our proposed approach centers around a groundbreaking Convolutional Neural Network design, comprising two parallel branches. The initial branch receives input from a distinctive shallow Local Phase Quantization (LPQ) invariant descriptor. This descriptor is derived from the amalgamation of color-texture information within the HSV color space. The HSV color space is particularly advantageous, as it effectively disentangles brightness from chromaticity, contributing to enhanced image stability. Conversely, the second branch of the CNN processes images originating from the RGB color space. This separation effectively isolates chromatic (color-related) details from achromatic (brightness-related) attributes. These two distinct CNN branches generate separate sets of deep feature vectors. Through the fusion of these resultant deep feature vectors based on the attention mechanism, we create an input vector for a subsequent Dense layer. This strategic concatenation empowers the system to discriminate between authentic, live faces and attempted spoofing instances. A graphical representation of our holistic approach is depicted in Figure 1.

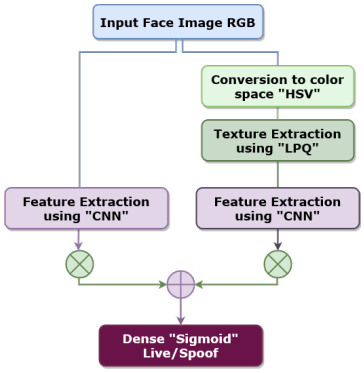

Fig. 1 Overall diagram of the proposed approach

Before feeding the two inputs of our CNN architecture (the two branches), with the input RGB images, firstly, we consider the MTCNN face detector [48] to extract a face robust region of interest from the input images to achieve an efficient feature extraction. Then, we use HSV [49] color space, combined with shallow LPQ [11] to extract spatial color and texture features from the input images, for the first branch. The choice of the HSV color space offers notable advantages in handling variations in illumination. The hue component, which primarily encodes color information, remains relatively stable under varying lighting conditions, making it well-suited for applications where lighting variations are common. The LPQ descriptor effectively captures local structural information, enabling it to identify unique and discriminative facial features. Compared to alternative feature extraction methods like local binary patterns (LBP), the LPQ descriptor is computationally efficient, involving simpler calculations and fewer operations. This efficiency makes it well-suited for real-time applications. While we maintain the RGB color space for the second branch. Both CNN branches continue to learn rich appearance features through their respective layers, yielding two vectors of deep features. These vectors are then concatenated to form an input vector for the subsequent Dense layer, facilitating the discrimination between live and spoofing faces. For a visual representation of our approach, refer to Figure 2, which illustrates the detailed architecture.

Our proposed approach encompasses three major steps: preprocessing, feature extraction, and classification. In the subsequent subsections, we will provide a comprehensive explanation of each of these steps.

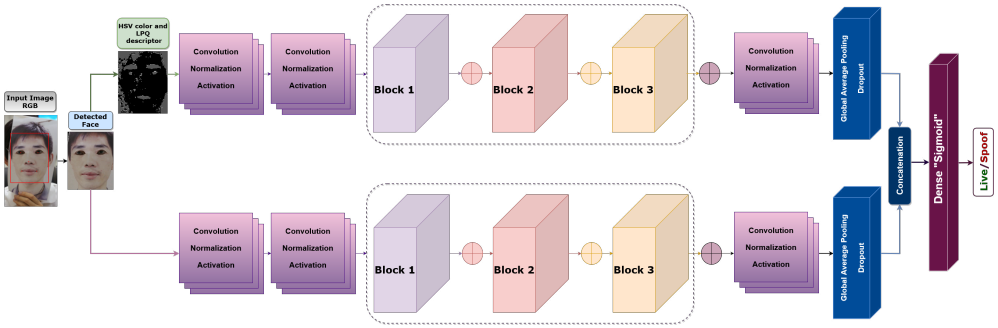

Fig. 2 Detailed architecture of the proposed approach

3.1 Face preprocessing

In the first stage, we use the Multi-Task Cascaded Convolutional Networks (MTCNN) algorithm [48] for face detection in the input images. Figures 3 and 4 provide a visual representation of the MTCNN algorithm for reference.

The MTCNN algorithm comprises four key components: **face classification**, **bounding box regression**, **landmark localization**, and **multi-source training**. In the first component, the Cross-Entropy loss function is utilized for face classification, where $y_i^{det} \in \{0, 1\}$ represents the ground-truth value, and p_i is the probability

of the input image being a face. In the second component, the Euclidean distance loss function is applied for bounding box regression to refine the location of the detected face. Here, $\dot{y}_i^{box}$ represents the predicted bounding box by the detector, while y_i^{box} corresponds to the ground truth object location coordinates. This bounding box is defined by four coordinates: left, top, height, and width. In the third component, which focuses on landmark localization, a similar approach to bounding box regression is employed, utilizing the Euclidean loss. Here, $\dot{y}_i^{landmark}$ is the coordinates of the facial landmarks, while $y_i^{landmark}$ represents the corresponding ground truth coordinates. The facial landmarks encompass five points, specifically the left eye, right eye, nose, left mouth, and right mouth locations. The fourth component pertains to the comprehensive learning objective for multiple image inputs during the training process. In this context, N stands for the number of training samples, α_j signifies the task importance, $\beta_i^j \in \{0, 1\}$ serves as the sample type indicator, and L_i^j denotes the aforementioned loss function. The formula encapsulating the MTCNN algorithm is expressed as follows (Eq. 1):

$$
\begin{aligned}
L_i^{det} &= -[y_i^{det} \times \log(p_i) + (1 - y_i^{det}) \times (1 - \log(p_i))] \\
L_i^{box} &= \|\dot{y}_i^{box} - y_i^{box}\|_2^2 \\
L_i^{landmark} &= \|\dot{y}_i^{landmark} - y_i^{landmark}\|_2^2 \\
\min \sum_{i=1}^{\mathrm{N}} & \sum_{j \in (det, box, landmark)} \alpha_j \times \beta_i^j \times L_i^j \\
\beta_i^j &\in \{0, 1\}
\end{aligned}
\tag{1}
$$

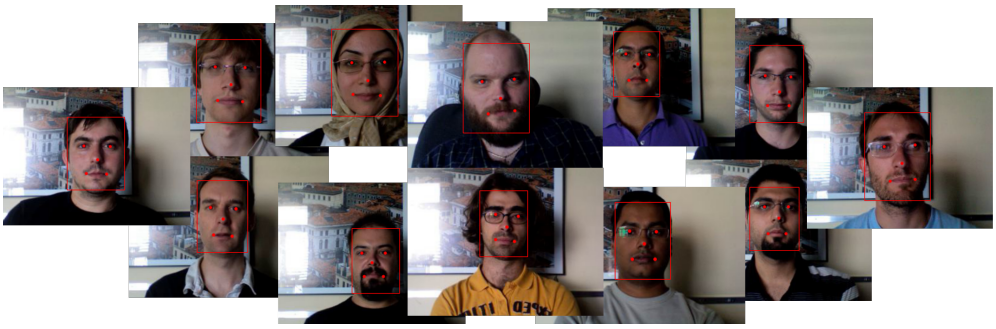

Fig. 3 An illustration of applying MTCNN algorithm on a sample images of the Replay-Attack dataset for faces detection

Fig. 4 An illustration of applying MTCNN algorithm on a sample images of the CASIA-FASD dataset for faces detection

3.2 Feature extraction algorithms

3.2.1 Hue-Saturation-Value (HSV) color space conversion

The initial step in the first branch of our proposed CNN approach involves the conversion of the facial image to the Hue-Saturation-Value (HSV) color space. HSV comprises three components in this cylindrical color model, as detailed in [49]. This model serves as a potent tool for enhancing image stability by segregating brightness from chromaticity. Hue represents various colors, Saturation quantifies the range of colors (the amount of gray), and Value signifies the level of lightness and darkness. The HSV color space proves invaluable for feature extraction, especially in the context of face spoofing detection. The conversion of image color space from RGB to HSV is accomplished using the following formulae (Eq. 2 to Eq. 5), as outlined in [50].

$$\mathrm{H} = \begin{cases} \mathrm{H_i} & \text{if } \mathrm{B} \leq \mathrm{G} \\ \\ 360 - \mathrm{H_i} & \text{if } \mathrm{B} > \mathrm{G} \end{cases} \tag{2}$$

$$\mathrm{H} = \cos^{-1}\left[\frac{0.5 \times [(R-G)+(R-B)]}{\sqrt{(R-G)^2 + (R-B) \times (G-B)}}\right] \tag{3}$$

$$S = \frac{\max(R, G, B) - \min(R, G, B)}{255} \tag{4}$$

$$V = \frac{\max(R, G, B)}{255} \tag{5}$$

3.2.2 Local Phase Quantization (LPQ) descriptor on HSV images

Following the conversion of the image color space to HSV, we employ the Local Phase Quantization (LPQ) texture descriptor, originally introduced by Ojansivu and Heikkil [11]. As depicted in Figure 5, our approach utilizes HSV color space in conjunction with the shallow LPQ texture descriptor for input images. LPQ is a hand-crafted operator known for its resilience against optical blurring, uniform illumination variations, misalignment, and its effectiveness in addressing expression variations. LPQ operates by conducting Short-Time Fourier Transform (STFT) calculations within a rectangular neighborhood at each pixel location, enabling the extraction of local phase information from the source image. This extracted local phase information is then encrypted, and LPQ estimates the distribution of these encrypted details to generate LPQ features. The mathematical formula for LPQ is elaborated below.

The image invariant spatial blur $p(m, n)$ is calculated using a convolution operation (Eq. 6):

$$q(m, n) = p(m, n) \otimes h(m, n) \tag{6}$$

Where $p(m, n)$ represents the original image, $q(m, n)$ represents the blurred image, $h(m, n)$ the point spread function (PSF), and $\otimes$ the convolution [51].

The Fourier representation of (Eq. 6) is provided by (Eq. 7):

$$Q(u, v) = P(u, v) \cdot H(u, v) \tag{7}$$

where $Q(u, v)$, $P(u, v)$, and $H(u, v)$ are the Fourier transforms of $q(m, n)$, $p(m, n)$, and $h(m, n)$.

Then, the phase information of the fuzzy image is obtained as follows (Eq. 8):

$$\angle Q(u, v) = \angle P(u, v) + \angle H(u, v) \tag{8}$$

where $\angle Q(u, v)$, $\angle P(u, v)$, and $\angle H(u, v)$ are the phases of $q(m, n)$, $p(m, n)$, and $h(m, n)$ respectively.

When the phase of the PSF, $h(m, n)$, is centrally symmetric, it has just two values and is represented by (Eq. 9):

$$\angle H(u, v) = \begin{cases} 0 & \text{if } H(u, v) \geq 0 \\ \\ \pi & \text{otherwise} \end{cases} \tag{9}$$

Therefore, the phase invariance between $Q(u, v)$ and $P(u, v)$ is given by (Eq. 10):

$$\angle Q(u, v) = \angle P(u, v), \qquad for\ all\ H(u, v) \geq 0 \tag{10}$$

Equation illustrates the estimation STFT calculated at each pixel point x of the MM neighborhood region of the image $q(m,n)$ defined by this formula (Eq. 11):

$$Q(u,v) = \sum_{m \in N_m} \sum_{n \in N_n} q(m,n) e^{-j2\pi(um+vn)/M} \tag{11}$$

where N_m and N_n denote the neighborhood region.

The local Fourier coefficients are calculated at four frequencies $Z_1 = (a,0)$, $Z_2 = (0,a)$, $Z_3 = (a,a)$, and $Z_4 = (a,-a)$, using STFT, where a is a very small integer that obeys (Eq. 10). The vector form of these points is presented as follows (Eq. 12):

$$V = \left[Q(z_1), Q(z_2), Q(z_3), Q(z_4)\right] \tag{12}$$

and (Eq. 13),

$$W = \left[Real\left\{V\right\}, Image\left\{V\right\}\right] \tag{13}$$

where $Real\left\{V\right\}$ and $Image\left\{V\right\}$ are the real and the imaginary components of V, respectively.

The resulting eight binary quantized coefficients k_i are represented as integer values between 0 and 225 using a binary encoding of the elements in W, given by (Eq. 14):

$$b_{LPQ} = \sum_{i=1}^{8} k_i 2^{i-1} \tag{14}$$

where k_i denotes the quantization of the i^{th} element in W as (Eq. 15):

$$k_i = \begin{cases} 1 & \text{if } W_i \geq 0 \\ \\ 0 & \text{otherwise} \end{cases} \tag{15}$$

Finally, the LPQ descriptor is obtained, providing a more precise representation of local features.

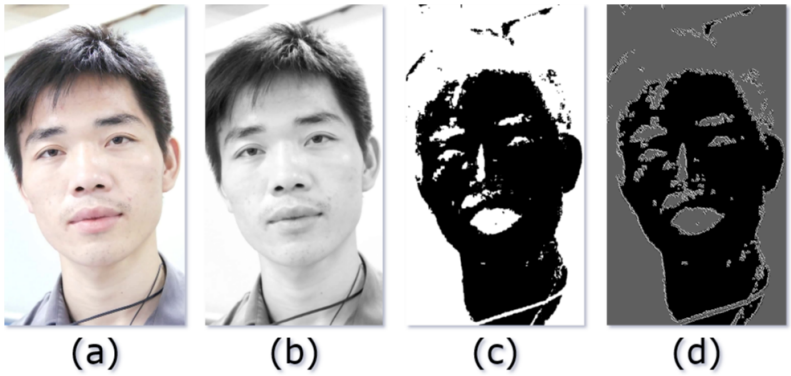

Fig. 5 Illustration of features extraction: (a) RGB original image, (b) Grayscale image, (c) HSV image, and (d) Output image in HSV color space and LPQ descriptor

3.2.3 Layer Convolutional Neural Network

The Convolutional Neural Network (CNN), often referred to as ConvNets, is a Deep Learning algorithm designed for processing data with a grid-like pattern, such as images. It incorporates a specialized deep layer structure within an Artificial Neural Network (ANN). The sequence of layers in our CNN-based approach is outlined below:

Firstly, the **Convolutional Layer** assumes a pivotal role within a CNN, tasked with the extraction of features from the input image (represented as a content matrix) via the utilization of kernels of varying sizes through a convolution operation. This operation entails the sliding of kernels across the input image, performing element-wise multiplications, and aggregating the outcomes to yield a feature map. By employing multiple kernels, this layer can capture a wide spectrum of features from the input image. The resultant feature volume following the application of the convolution operation can be determined through the mathematical expression provided by Eq. (16).

$$\begin{aligned} \mathrm{V_{out}} &= \mathrm{W_{out}} \times \mathrm{H_{out}} \times \mathrm{C} \\ \mathrm{W_{out}} &= \left[\frac{\mathrm{W_{in}} - \mathrm{K} + 2 \times \mathrm{P}}{\mathrm{S}}\right] + 1 \\ \mathrm{H_{out}} &= \left[\frac{\mathrm{H_{in}} - \mathrm{K} + 2 \times \mathrm{P}}{\mathrm{S}}\right] + 1 \end{aligned} \tag{16}$$

In this context, W_{in} and H_{in} represent the width and height of the input image, which is of size (W × H × L). K denotes the size of the kernel, P is the padding, S is the stride, C is the number of kernels applied, and V_{out} the volume of the output image.

Following the convolutional layer, a **Batch-Normalization Layer** is employed to normalize the output feature maps from each layer. This normalization process accelerates training and improves the overall learning procedure. The mathematical expression for the batch normalization layer is defined in Eq. (17).

$$Y_i \leftarrow \gamma \dot{X}_i + \beta$$

$$\dot{X}_i \leftarrow \frac{X_i - \mu_b}{\sqrt{\sigma_b^2 + \epsilon}}$$

$$\mu_b \leftarrow \frac{1}{m} \sum_{i=1}^{m} X_i \tag{17}$$

$$\sigma_b \leftarrow \sqrt{\frac{1}{m} \sum_{i=1}^{m} (X_i - \mu_b)^2}$$

where Y_i is the accumulated average normalization, while γ and β are the parameters for learnable scale and shift, respectively. The constant ϵ is included for stability purposes, m refers to the batch size, μ_i represents the mean, σ_i is the standard deviation, and $\dot{X}_i$ corresponds to the input normalization of the layer.

After the Batch-Normalization layer, the **Swish Activation Function** [52] is applied to normalize data and enable faster convergence and learning. It outperforms ReLU [53] and is a special case of the Sigmoid ($f(\beta x)$ Eq. 23) shrinkage function. The Swish function is defined as follows Eq. (18).

$$\text{Swish(x)} = \text{x} \times f(\beta x) \tag{18}$$

where β is a trainable parameter. When $\beta = 0$, Swish is equivalent to a linear function Swish(x) = x, and when $\beta \to \infty$, the Swish can be approximated as a ReLU function when the Sigmoid approaches a 0 and 1 function.

Next, the **Average Pooling Layer** is employed to decrease the number of training parameters and computational workload. It achieves this by computing the average value over a group of neurons in the preceding layer, resulting in an average output for each feature map. This operation is instrumental in capturing robust and invariant features, simultaneously diminishing the CNN's sensitivity to minor input variations. The mathematical representation of the average pooling operation is elucidated by Eq. (19).

$$M_j = \tanh(\beta \sum_{N \times N} M_i^{n \times n} + \alpha) \tag{19}$$

where M represents the inputs to the average pooling layer, β is a trainable scalar, $M_i^{n \times n}$ is a sub-matrix of the averages of M, and α is a bias.

Furthermore, the **Global Average Pooling Layer** conducts average pooling calculations on the feature maps produced by the final convolutional layer in each

branch. This operation yields a single value for each feature map, effectively reducing the dimensionality of the feature maps while retaining the most critical information. Subsequently, these resulting values are concatenated into a unified output vector. The mathematical representation of the global average pooling operation has been presented earlier in Eq. (20).

$$G_{avg} = \frac{1}{m} \sum_{i=1}^{m} \chi^{l}_{1:h,1:w,i} \tag{20}$$

Here, G_{avg} represents the result obtained from the global average pooling operation. l denotes the output index, m stands for the total number of element values in the kernel, and the ranges denoted by $1:h$ and $1:w$ in the height and width directions encompass the first line to the h^{th} line and the first column to the w^{th} column, respectively. h and w correspond to the height and width, while χ is the element value corresponding to the filter.

The **Dropout Layer** is a regularization technique primarily employed in tandem with Global Average Pooling layers to mitigate the issue of overfitting within each branch. This method entails the random and temporary deactivation of a portion of neurons during the training phase, effectively disregarding the outputs of these neurons. The mathematical representation of a dropout operation is expressed as shown in Eq. (21).

$$\begin{aligned} Y_i^{l+1} &= f(Z_i^{l+1}) \\ Z_i^{l+1} &= W_i^{l+1} \odot Y^l + Bi^{l+1} \end{aligned} \tag{21}$$

where Y_i^{l+1} is the final activation output value of the i^{th} neuron of the $(l+1)^{th}$ layer, Y^l is the input value of the intermediate activation of the l^{th} layer before dropout, f is the swish activation function, Z_i^{l+1} is the linear combined output value of the i^{th} neuron of the $(l+1)^{th}$ layer, W_i^{l+1} is the weight value of the i^{th} neuron of the $(l+1)^{th}$ layer, i is the i^{th} neuron, l and $l+1$ are the l^{th} and $(l+1)^{th}$ layers, and B^l is the bias of the l^{th} layer.

Ultimately, a **Dense Layer** featuring a Sigmoid activation function is employed to differentiate between live and spoofing faces and make the final classification decision. This layer constitutes a fully connected artificial neural network (ANN) layer, wherein every neuron establishes connections with the preceding layer. When the input values are denoted as x_i (for $i = 1, 2, \ldots, n$), the output is represented as Y, and the relationship between Y and x_i is depicted in Eq. (22). The architecture of the Dense and Sigmoid layers is depicted in Figure 6.

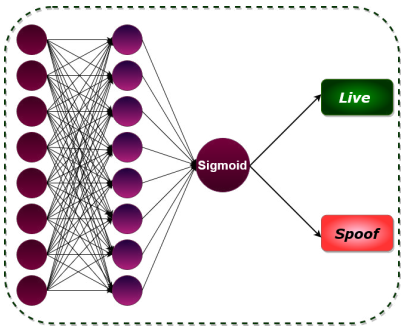

Fig. 6 Illustrates the output Dense Layer

$$\mathrm{Y} = f(\sum_{\mathrm{i}=1}^{\mathrm{n}}(\mathrm{w_i} \times \mathrm{x_i} + b)) \tag{22}$$

where $f(.)$ is the Sigmoid activation function, w_i and b are the connection coefficient and the additive deviation, respectively.

The output of the dense layer is then fed into the **Sigmoid** [54] (f) output layer, which transforms any real value into a value between 0 and 1 using a non-linear logistic function, as defined in Eq. (23).

$$f(\beta x) = \frac{1}{1 + e^{-\beta x}} \tag{23}$$

where x is the output value of the neuron.

3.3 Architecture of the proposed approach and classification

In this section, we introduce a novel CNN architecture based on a modified and reduced version of the Xception model, introduced by Chollet [55], for deep feature extraction. The basic Xception model is a deep CNN architecture that uses depth-wise separable convolutions [55]. It is an architecture employed in various domains of image processing and computer vision [56, 57]. The Xception architecture was created using 36 convolutional layers that form the feature extraction foundation of the Xception network. The Xception network with a convolutional base accompanied by a logistic regression layer can be utilized for facial image classification. The fully connected (FC) layers of the network are added beforehand with the logistic regression layer. The 36 convolutional layers of the network are assembled into 14 modules that have a linear residual connection with the exception of the start and end modules. In this network, the initial data flows through the entry flow block, then the data flows through the middle flow block, which iterates 8 times, and finally, the data flows through the exit flow. Batch normalization is applied after both the convolution and separable convolution layers. The transition layer includes a 1 $\times$ 1 convolution layer and 2 $\times$ 2 maximum pooling layers. The feature map sizes remain consistent within the dense block, facilitating seamless combination. Subsequently, global average pooling is implemented

after the last dense block in the network. The classification of facial images is carried out using the Softmax classifier integrated into the network. The basic Xception model offers several advancements compared to earlier CNN architectures, such as increased depth and a larger number of parameters. This enhanced model capacity allows the network to effectively capture intricate patterns and representations in the data, resulting in improved accuracy and performance. Additionally, it integrates skip connections to facilitate the direct propagation of information across different network layers. By exploiting these skip connections, the network can effectively capture local and global dependencies, enabling a seamless flow of information and better gradient propagation during training.

Our proposed CNN architecture follows a structured design comprising two branches, each with the same architecture but different inputs. In the first branch, we employ a combination of shallow LPQ features and the HSV color space. Initially, we convert the input images to the HSV color space and subsequently compute the LPQ descriptor for each image. For the second branch, we retain the original RGB color space of the input images, which includes three color channels: red, green, and blue. This approach separates chromatic (color-related) information from achromatic (brightness-related) information, enabling the extraction of crucial features. Both CNN branches have an identical architecture, structured as follows: 1) Two sequential sets of layers, each containing a convolutional layer, a batch normalization layer, and a swish activation function. These two sets of layers facilitate the reception of processed image data and its transmission to the subsequent layer. 2) Three identical blocks of layers, each composed of two parallel sub-branches. The first sub-branch comprises only a convolutional layer, while the second sub-branch consists of two identical sub-blocks (convolution, batch normalization, and swish activation function) followed by an average pooling layer. Finally, an addition layer combines the outputs of the average pooling layer and the convolutional layer from the first sub-branch. 3) After the three blocks, there is a convolutional layer, batch normalization layer, swish activation function, global average pooling layer, and a dropout layer. The two principal CNN branches are finally merged using a concatenate layer, which combines their respective outputs. This combined feature set is then fed into the final Dense layer, a fully connected layer that processes the merged features to generate a single output vector. This output vector is subsequently passed through a sigmoid activation function to make the final classification decision, distinguishing live faces from spoofed faces. The comprehensive CNN architecture is depicted in Figure 7.

The proposed approach attention mechanism based fusion method can be formulated as follows (Eq. 24): $f_{(HSV-LPQ)}$ and f_{RGB} represent the extracted features from the CNN branch extractors, which generate two output vectors. These output vectors include scores for each feature, represented by (fi, i = 1, ..., n). Additionally, (ω_i, i = 1, ..., n) corresponds to the set of weights assigned to each feature. The fusion function, denoted as $\mathbb{F}$, efficiently combines the extracted feature vectors from $f_{(HSV-LPQ)}$ and f_{RGB}, resulting in a fused feature vector, ν. This fused feature vector comprehensively captures the combined information from two $f_{(HSV-LPQ)}$ and f_{RGB}, thereby providing a more robust representation for the model.

$$\nu = \mathbb{F}(f_{(HSV-LPQ)}, f_{RGB}) = \sum_{i=1}^{n} \omega_i \otimes fi \tag{24}$$

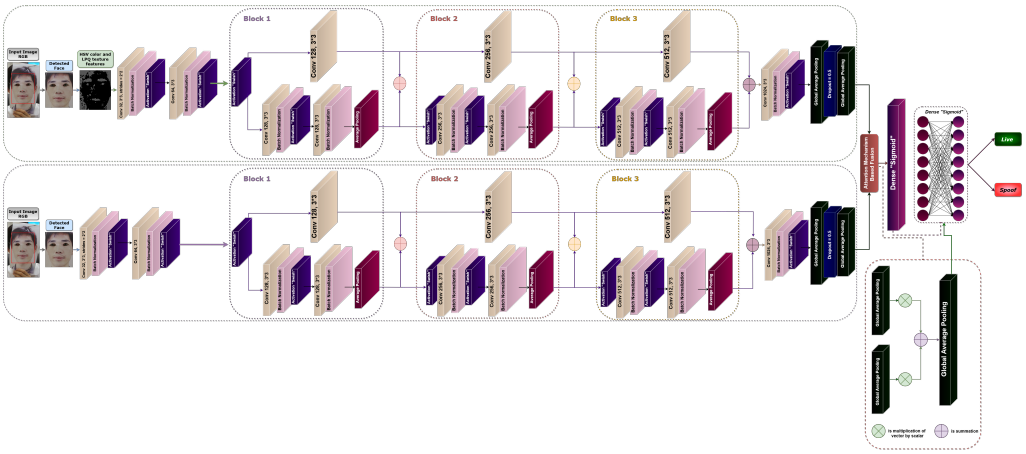

Fig. 7 Typical blocks of the CNN architecture

4 Experimental results

In this section, we provide a comprehensive overview of the experimental evaluation and the results obtained from our approach for face spoofing detection. Details concerning the experiments and the databases used are further expounded upon in the following subsections.

4.1 Dataset

We evaluate our approach based of a series of experiments on two public benchmark databases called CASIA-FASD [5] and Replay-Attack [13].

CASIA-FASD is the first publicly available Face Anti-Spoofing Database that provides three types of attacks: warped printed photos, printed photos with cut eyes, and video replay attacks for each subject. It contains 50 subjects divided into 20 subjects for the training set and 30 subjects for the testing set. Each subject includes three different lighting conditions: low, middle, and high-quality. Example samples of the CASIA-FASD dataset are shown in Figure 8.

Fig. 8 Live faces and Spoofing faces (Warped photo, Cut photo, and Video replay attacks) examples from the CASIA-FASD dataset

Replay-Attack Database for face spoofing consists of 1200 videos from 50 subjects. This dataset is split into 15 subjects for training with 360 videos, 15 subjects for development with 360 videos, and 20 subjects for testing with 480 videos. The dataset divides the attacks into two types: printed photo and video replay attacks. The dataset also divides the attacks into two types of holding conditions: hand-held and stand-fixed, under two different lighting conditions: controlled and adverse. The training subset is utilized to train the countermeasure model, the development set is utilized to fine-tune the model, and the testing subset is utilized to evaluate the performance. Example samples of the Replay-Attack dataset are shown in Figure 9.

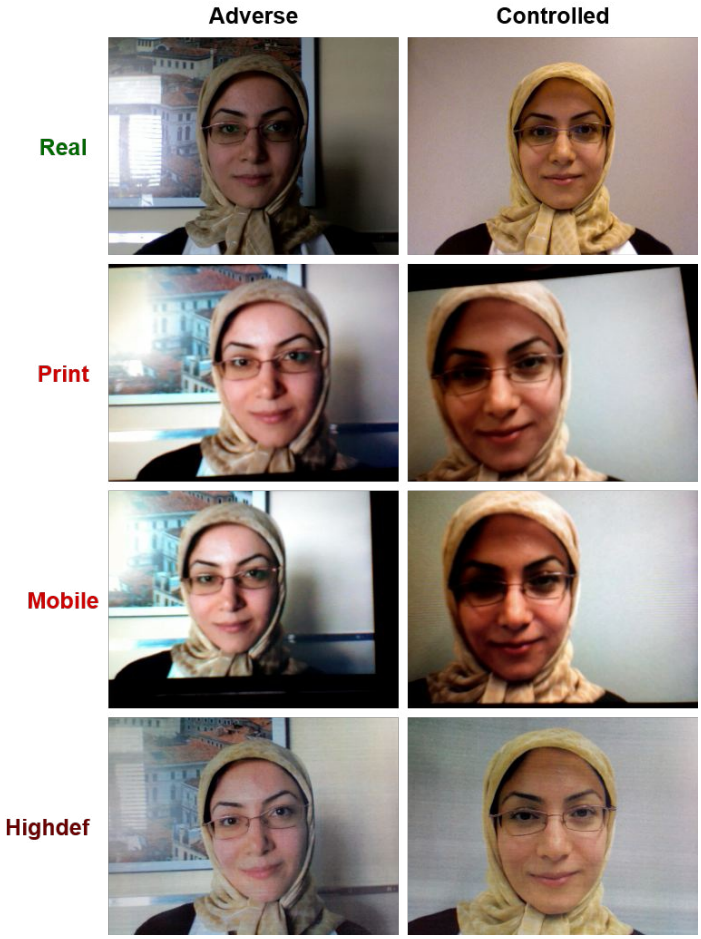

Fig. 9 Live faces and Spoofing faces (high definition, mobile and print attacks) in different scenarios (adverse and controlled), examples from the REPLAY-ATTACK database

4.2 Implementation and experimental evaluation

Our approach (see Section 3) is implemented in Python language using Keras-TensorFlow deep learning framework [58]. The parameters of our approach are as follows: The Dropout layer is set to a rate of 0.5 to prevent over-fitting and we set the number of epochs as 15. The CASIA-FASD dataset is originally organized into two subsets: training and test. In our experiments, we consider 10% of the training subset as validation subset and the rest of training subset (90%) as the new training subset. The Replay-Attack dataset is originally organized into three subsets: training, validation, and testing. Our approach was evaluated using 5-fold cross-validation: the original CASIA-FASD and Replay-Attack datasets were randomly divided into five

equal-sized subsamples. Out of the five subsamples, one was held out as validation data to test the model, and the remaining four subsamples were used as training data. The 5-fold cross-validation process was repeated five times, with each of the five subsamples being used exactly once as validation data. The five results were averaged to produce a single estimate. For the training process, we consider the Binary Cross-Entropy loss function [59], and the optimization algorithm Adam [60] to minimize the loss function with the learning rate 1×10^{-3}.

The Loss function of Binary Cross-Entropy is a sum of two losses, namely the classification loss and the regression loss given by Eq. (25),

$$\text{Binary\_Cross\_Entropy} = -\frac{1}{\text{N}}\sum_{i=1}^{\text{N}}[\rho_i \times \log(\gamma_i) + (1-\rho_i) \times \log(1-\gamma_i)] \tag{25}$$

Where N is the number of training samples, i is the index of training sample, ρ_i is the predicted label of the sample, and γ_i is the value actual label of the sample.

The Adaptive Moment Estimation (Adam) algorithm is based on Stochastic Gradient Descent (SGD) optimization, which is outlined in the following equation (Eq. 26), evaluates and accumulates expectations for both the gradient and its second moment for each iteration.

$$\begin{aligned}
\bigtriangledown &\leftarrow \bigtriangledown_\theta \sum_{t=1}^{T} f_t(\theta) \\
\text{t} &\leftarrow \text{t} + 1 \\
g &\leftarrow \beta_1 g + (1-\beta_1)\bigtriangledown \\
r &\leftarrow \beta_2 r + (1-\beta_2)\bigtriangledown \odot \bigtriangledown \\
\dot{g} &\leftarrow \frac{g}{1-\beta_1^{\text{t}}} \\
\dot{r} &\leftarrow \frac{r}{1-\beta_2^{\text{t}}} \\
\theta &\leftarrow \theta - \frac{\alpha \cdot \dot{g}}{\sqrt{\dot{r}} + \delta}
\end{aligned} \tag{26}$$

where $f_1(\theta)$,..., $f_T(\theta)$ is a stochastic scalar function at subsequent timesteps 1,..., T, $\bigtriangledown$ is a vector of partial derivatives of f_t, with respect to θ evaluated at timestep t, g is the estimate of the accumulated gradient, r is the estimate of the accumulated

raw moment, β_1 and β_2 are the hyper-parameters of the exponential decay rates while α and δ are stability parameters.

We report our results using the following performance metrics: Equal Error Rate (EER), False Acceptance Rate (FAR), False Rejection Rate (FRR), Half Total Error Rate (HTER), Attack Presentation Classification Error Rate (APCER), Bona Fide Presentation Classification Error Rate (BPCER), Average Classification Error Rate (ACER), Accuracy, Precision, Recall, F1-Score and False Negative Rate (FNR).

The formulas of the metrics are given as follow (Eqs. 27, 28, 29, 30, 31, 32, 33, 34 and 35 respectively).

$$\text{EER} = \text{FAR} - \text{FRR} \tag{27}$$

$$\text{HTER} = \frac{\text{FAR} + \text{FRR}}{2} \tag{28}$$

$$\text{APCER} = \frac{\text{FP}}{\text{TN} + \text{FP}} \tag{29}$$

$$\text{ACER} = \frac{\text{APCER} + \text{BPCER}}{2} \tag{30}$$

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FN} + \text{TN} + \text{FP}} \tag{31}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{32}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{33}$$

$$\text{F}_1 - \text{Score} = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \tag{34}$$

$$\text{FNR} = \frac{\text{FN}}{\text{FN} + \text{TP}} \tag{35}$$

Where, TP, TN, FN, FP represents the number of true positive, true negative, false negative, false positive respectively, the confusion matrix is as Table 1.

Table 1 The Confusion Matrix

Actual	Predict	
	Live Face	Spoof Face
Live Face	Ture Positive (TP)	False Negative (FN)
Spoof Face	False Positive (FP)	True Negative (TN)

4.3 Results and discussions

In this section, we showcase the outcomes of our face spoofing detection approach, evaluated using different performance metrics applied to the CASIA-FASD and Replay-Attack datasets. As delineated in Tables 2, 3, 4, and 5 (across the various metrics considered), we facilitate comparisons between our approach and the current state-of-the-art methods. These comparisons are based on different performance metrics, providing a comprehensive evaluation on the CASIA-FASD and Replay-Attack databases, which we elaborate on below.

Table 2 shows the experimental results of our approach compared to the state-of-the-art methods (hand-crafted (see Subsection 2.1) and CNN methods (see Subsection 2.2)) for the three evaluation metrics ACER, APCER, and FNR. Our approach achieved an ACER of 0.04% and 0.05%, an APCER of 0.03% and 0%, and a FNR of 0.06% and 0.11%, for the CASIA-FASD and Replay-Attack datasets, respectively. These results surpassed other CNN methods, for example, Guo et al. [37] with an ACER of 11.72%, and an APCER of 4.68% on the CASIA-FASD dataset, and VGG-16 [22] with an FNR of 4.29% and 8.18% for the CASIA-FASD and Replay-Attack datasets, respectively.

Table 2 Comparison of the results obtained by our proposed approach and state-of-the-art on CASIA-FASD and Replay-Attack datasets with different metrics (ACER (%), APCER (%), and FNR (%))

Methods & Datasets	CASIA-FASD			Replay-Attack		
	ACER	APCER	FNR	ACER	APCER	FNR
Our Approach	**0.04**	**0.03**	**0.06**	**0.05**	**0**	**0.11**
Guo et al. [37]	11.72	4.68	3.43	/	/	/
Abdullakutty et al. VGG-16 [22]	/	/	4.29	/	/	8.18

Tables 3 and 4 presents a comparison of results between our proposed approach and state-of-the-art CNN-based architectures (refer to Subsection 2.2) on the CASIA-FASD and Replay-Attack databases, using four evaluation metrics: Accuracy, Precision, Recall, and F1-Score. Our approach demonstrates superior performance across all metrics when compared to other methods. The results highlight our approach's performance with an Accuracy of 96.00% and 97.00%, Precision of 97.00% and 97.00%, Recall of 97.00% and 98.00%, and F1-Score of 96.00% and 97.00% for the CASIA-FASD and Replay-Attack datasets, respectively. The alternative CNN-based architectures, such as VGG-16 [22], achieved lower scores with an Accuracy of 85.00% and 84.00%, Precision of 87.00% and 88.00%, Recall of 96.00% and 92.00%, and F1-Score of 91.00% and 90.00% for the CASIA-FASD and Replay-Attack datasets, respectively. Similarly, Xception [25] also obtained a lower score with an Accuracy of 62.00% on the CASIA-FASD dataset.

Table 3 Comparison of the results obtained by our proposed approach using different metrics (Accuracy (%), Precision (%), Recall (%), and F1-Score (%)) with state-of-the-art tested on CASIA-FASD dataset

Methods & Datasets	CASIA-FASD			
	Accuracy	Precision	Recall	F1-Score
Our Approach	**0.96**	**0.97**	**0.97**	**0.96**
Satapathy et al. Inception-V2 [24]	0.94	0.95	0.92	0.93
Abdullakutty et al. ResNet-50 [22]	0.93	0.92	/	/
Abdullakutty et al. DenseNet-121 [23]	0.93	/	/	/
Satapathy et al. ResNet-34 [24]	0.92	0.94	0.90	0.91
Satapathy et al. ResNet-18 [24]	0.92	0.94	0.90	0.91
Satapathy et al. ResNeXt-50 [24]	0.91	0.92	0.90	0.90
Satapathy et al. GoogleNet [24]	0.88	0.92	0.83	0.87
Abdullakutty et al. Inception-V3 [22]	0.86	0.86	0.97	0.92
Abdullakutty et al. VGG-16 [22]	0.85	0.87	0.96	0.91
Satapathy et al. VGG-19 [24]	0.83	0.88	0.77	0.82
Satapathy et al. AlexNet [24]	0.83	0.85	0.81	0.82
Gwyn et al. Xception [25]	0.62	/	/	/

Table 4 Comparison of the results obtained by our proposed approach using different metrics (Accuracy (%), Precision (%), Recall (%), and F1-Score (%)) with state-of-the-art tested on Replay-Attack dataset

Methods & Datasets	Replay-Attack			
	Accuracy	Precision	Recall	F1-Score
Our Approach	**0.97**	**0.97**	**0.98**	**0.97**
Abdullakutty et al. ResNet-50 [22]	0.95	0.95	/	/
Abdullakutty et al. DenseNet-121 [23]	0.95	/	/	/
Abdullakutty et al. Inception-V3 [22]	0.88	0.88	0.98	0.93
Abdullakutty et al. VGG-16 [22]	0.84	0.88	0.92	0.90

In Table 5, we present the experimental outcomes of our approach, compared to state-of-the-art techniques, which encompass hand-crafted methods, CNN-based approaches, and hybrids fusing hand-crafted with CNN methods (as detailed in Section 2), all evaluated on the CASIA-FASD and Replay-Attack datasets. Remarkably, our approach excels in these evaluations, surpassing other methods with exceptional performance. We reached an EER of 0% and 0%, signifying a perfect discrimination, and a tiny HTER of 0.05% and 0.01%, for the CASIA-FASD and Replay-Attack datasets, respectively. These results underscore the effectiveness of our approach in face spoofing detection. While fusion is Hand-crafted with CNN methods, namely khammari. [39] achieved an EER of 2.62% and 0.53%, and an HTER of 2.14% and 0.69%, Atoum et al. [41] achieved an EER of 2.67% and 0.79%, and an HTER of 2.27% and 0.72% for the CASIA-FASD and Replay-Attack datasets, respectively. The CNN method, where Guo et al. [37] achieved an EER of 2.22% and 0.25%, and an HTER of 1.67% and 0.63% for the CASIA-FASD and Replay-Attack datasets, respectively. Among the hand-crafted methods, Boulkenafet et al. (HSV + LPQ) [15] obtained an EER of 7.40% on the CASIA-FASD dataset, and an EER of 7.90% and a HTER of 9.20%

on the Replay-Attack dataset using the LPQ descriptor where the features extracted from the HSV color space improve the performance, compared to Boulkenafet et al. LPQ [15] to their RGB scale counterparts which obtained an EER of 14.4% on the CASIA-FASD dataset, and an EER of 9.7% and a HTER of 10.3% on the Replay-Attack dataset. The results demonstrate the effectiveness of combining hand-crafted with CNN methods for face anti-spoofing.

Table 5 Comparison of the results obtained by our proposed approach with existing methods on CASIA-FASD and Replay-Attack datasets using different metrics (EER (%) and HTER (%))

Methods & Datasets	CASIA-FASD		Replay-Attack	
	EER	HTER	EER	HTER
Our Approach	**0**	**0.05**	**0**	**0.01**
khammari. [39]	2.62	2.14	0.53	0.69
Atoum et al. [41]	2.67	2.27	0.79	0.72
Antil et al. [45]	2.37	3.20	0	0
Shu et al. [47]	2.90	/	4.70	0.39
Kong et al. SE-ResNet50-Attention [33]	2.02	1.84	0.20	0.02
Guo et al. [37]	2.22	1.67	0.25	0.63
Sun et al. [32]	2.50	/	0.2	/
Boulkenafet et al. [15]	2.10	/	0.40	2.80
Boulkenaf et al. [16]	2.80	/	0.10	2.20
Chen et al. Attention [42]	3.14	/	0.21	0.38
Souza et al. [31]	4.44	/	0.33	2.50
Partial CNN [27]	4.50	/	2.90	6.10
Fine-tuned VGG-Face [27]	5.20	/	8.40	4.30
Antil et al. Modifed Xception [45]	7.45	5.08	2.45	1.50
Boulkenafet et al. (HSV + LPQ) [15]	7.40	/	7.90	9.20
Asim et al. [44]	8.02	9.94	3.22	4.70
Wang et al. MobileNetV2 [26]	9.40	16.7	3.6	12.6
Wang et al. ShuffleNetV2 [26]	14.9	21.9	6.33	21.8
Abdullakutty et al. VGG-16 [22]	/	24.01	/	23.05
Abdullakutty et al. DenseNet-121 [23]	/	12.85	/	6.76
Abdullakutty et al. ResNet-50 [23]	/	13.61	/	8.61
Boulkenafet et al. LPQ [15]	14.4	/	9.7	10.3
Abdullakutty et al. InceptionV3 [23]	/	27.16	/	29.43
Chingovska et al. [13]	/	18.2	/	13.8

As previously mentioned, we employed a 5-fold cross-validation for both the CASIA-FASD and Replay-Attack datasets. The performance of our approach on the 5-folds, measured in terms of Accuracy (%), Precision (%), Recall (%), and F1-Score (%), using the K-fold method is detailed in Table 6. Notably, the F1-score of our proposed approach surpasses 98.00% on the Replay-Attack dataset, showcasing remarkable performance. Furthermore, Table 6 illustrates that the Precision (%) and Recall (%) of our proposed approach are consistent across both the CASIA-FASD and Replay-Attack datasets. Particularly noteworthy is the accuracy of our approach in predicting nearly all images, as depicted in Figure 13. This observation underscores the efficiency and reliability of our approach in accurately and precisely distinguishing between real and spoofed faces, thus demonstrating its outstanding performance.

Table 6 Performance analysis of our approach using the K-fold method with different metrics (Accuracy (%), Precision (%), Recall (%), and F1-Score (%)), tested on CASIA-FASD and Replay-Attack datasets

Datasets	CASIA-FASD				Replay-Attack			
K-fold	Acc	Precision	Recall	F1-Score	Acc	Precision	Recall	F1-Score
1	0.96	0.97	0.97	0.97	0.96	0.97	0.97	0.99
2	0.96	0.98	0.97	0.96	0.97	0.98	0.97	0.98
3	0.96	0.96	0.97	0.96	0.98	0.96	0.97	0.97
4	0.96	0.98	0.98	0.95	0.98	0.98	0.98	0.98
5	0.97	0.97	0.98	0.96	0.98	0.97	0.98	0.98
Average	**0.96**	**0.97**	**0.97**	**0.96**	**0.97**	**0.97**	**0.97**	**0.98**

The Confusion Matrices depicted in Figure 10, the Receiver Operating Characteristic (ROC) curves in Figure 11, and the Precision-Recall curves in Figure 12 collectively validate the high performance of our approach. Additionally, Figure 13 provides examples of predicted results, further illustrating the effectiveness of our approach.

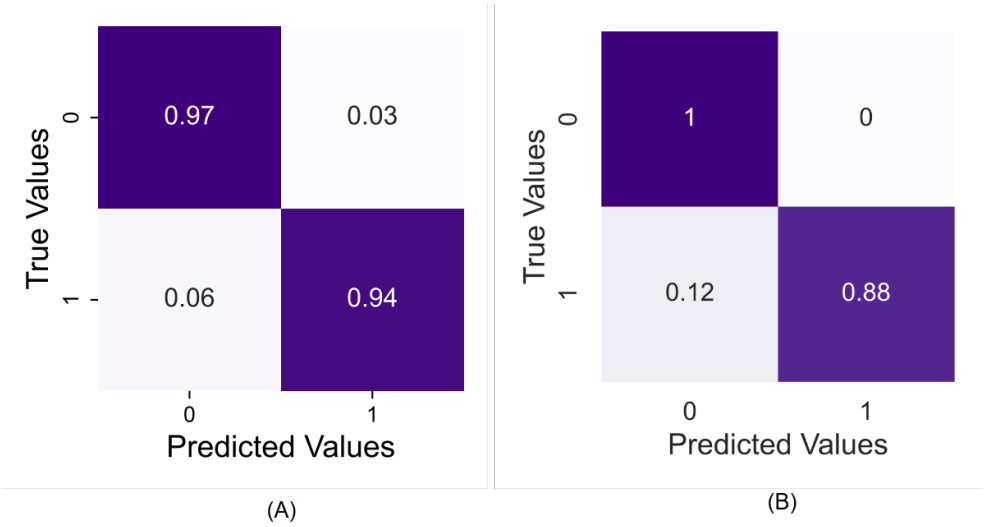

Fig. 10 Classification performance with Confusion Matrix using: (A) the CASIA-FASD dataset, and (B) the Replay-Attack dataset

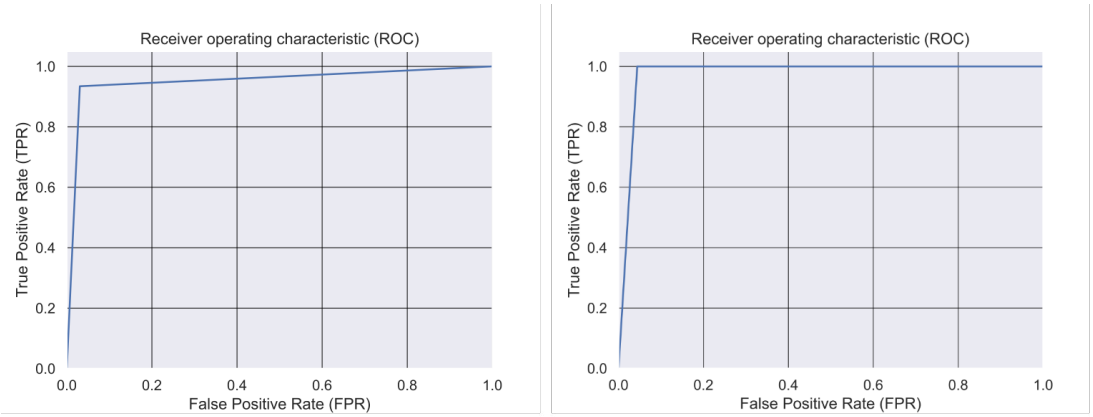

Fig. 11 The ROC curve using: (C) the CASIA-FASD dataset, and (D) the Replay-Attack dataset

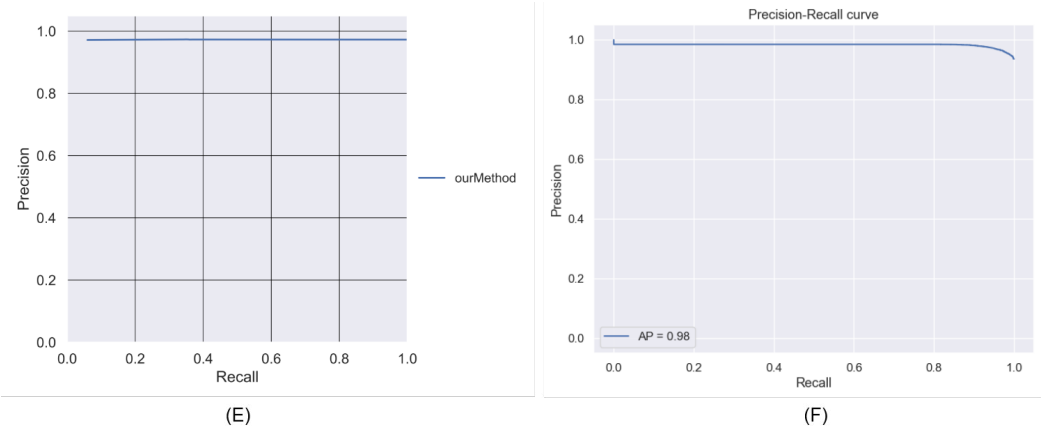

Fig. 12 The Precision-Recall using: (E) the CASIA-FASD dataset, and (F) the Replay-Attack dataset

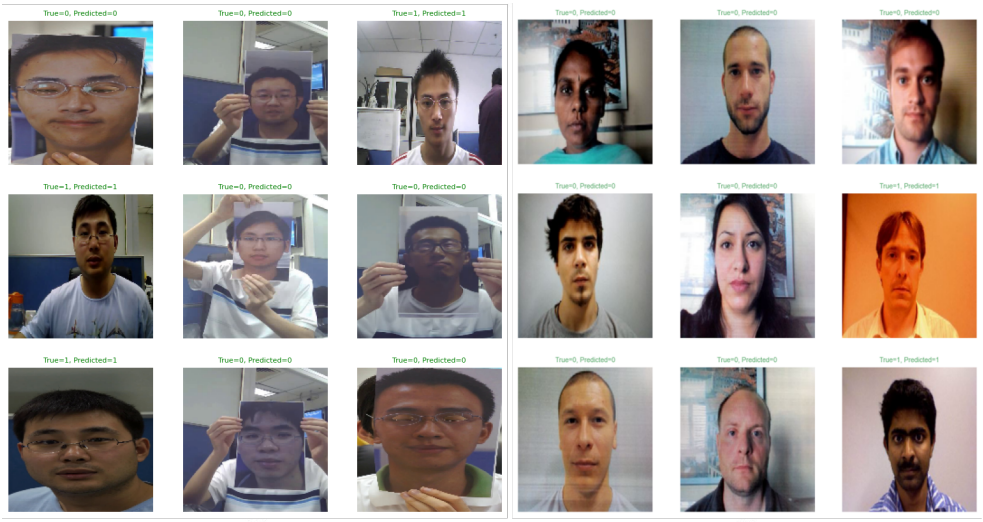

Fig. 13 Examples of live and spoof samples with their predictions from: (G) CASIA-FASD dataset, and (H) Replay-Attack dataset

4.4 Ablation study

Let remind that our proposed CNN architecture follows a structured design comprising two branches, each is based on a modified and reduced version of the Xception model [55]. Indeed, the two input sources (RGB and HSV-LPQ) of each branch can learn different patterns in the data. This can enable the model to capture significant and complementary information in both color spaces. Specifically, we reduced and modified the original Xception architecture namely by adding, deleting, and/or replacing a subset of original Xception layers, such as the layers: separable convolution, relu activation, max pooling, and logistic regression, by the layers: convolution, swish activation, average pooling, and sigmoid. Table 7 presents the layers utilized for the baseline Xception model [55] and our approach model. Figure 14 shows the architecture of our approach CNN branch and Xception baseline model [55]. Actually, the fusion of these two outputs from the two branches, each containing modified Xception, with the attention mechanism offers a powerful approach to improve data representation, manage complex interactions, and increase the adaptability of our CNN architecture (the two branches), which can lead to better overall performance in face spoofing attack detection. To validate the proposed approach in terms of fusion, our approach relies on the attention mechanism, compared to concatenation (Antil et al. [45]) and weighted average (Atoum et al. [41]). Table 8 shows that the proposed approach far outperforms traditional fusion methods.

The results from our ablation study reveal that the modifications applied to our two-branch CNN architecture, based on the modified Xception, significantly enhance its performance on the CASIA-FASD and Replay-Attack datasets. In comparison to the Xception baseline model [55], our approach (modified Xception with two branches)

outperforms the Xception baseline [55] with an EER of 0% and HTER of 0.05% on the CASIA-FASD dataset, and an EER of 0% and HTER of 0.01% on the Replay-Attack dataset. This improvement is attributed to the structure's ability to detect traces of facial identity spoofing, and the simultaneous use of two branches within the modified Xception architecture allows for the attainment of optimal results. Table 9 provides a detailed presentation of the results obtained for the baseline model [55] and our approach on the CASIA-FASD and Replay-Attack datasets.

These results are due to several factors, including:

- Using a standard convolution rather than a separable convolution allows for more information to be passed to the next layer, which can improve the accuracy of our CNN architecture.
- Using the swish activation rather than the relu activation can lead to improved energy efficiency of our CNN architecture and contribute to faster convergence and more stable optimization.
- Using the average pooling rather than the max pooling significantly enhances the robustness of our CNN architecture to noise in the data and provides better translation invariance.
- Using the sigmoid rather than the logistic regression allows for improved accuracy of our CNN architecture by reducing the number of parameters to be tuned.

This demonstrates the superiority of our approach compared to the baseline Xception model [55]. The results of our study suggest that our approach is a promising foundation for face spoofing attack detection.

Table 7 Number of layers in the baseline Xception model and the modified Xception model

Layer	Model/Number of layers	
	Modified Xception	Baseline Xception
Conv 1x1, stride=2x2	—	4
Conv 32, 3x3, stride=2x2	1	1
Conv 64, 3x3	1	1
Conv 128, 3x3	3	—
Conv 256, 3x3	3	—
Conv 512, 3x3	3	—
Conv 1024, 3x3	1	—
Depthwise Separable Conv 128, 3x3	—	2
Depthwise Separable Conv 256, 3x3	—	2
Depthwise Separable Conv 728, 3x3	—	6
Depthwise Separable Conv 1024, 3x3	—	1
Depthwise Separable Conv 1536, 3x3	—	1
Depthwise Separable Conv 2048, 3x3	—	1
Swish	9	—
ReLU	—	14
Average Pooling	3	—
Max Pooling layer, 3x3, stride=2x2	—	4
Global Average Pooling	1	1
Sigmoid	1	—
Logistic regression	—	1

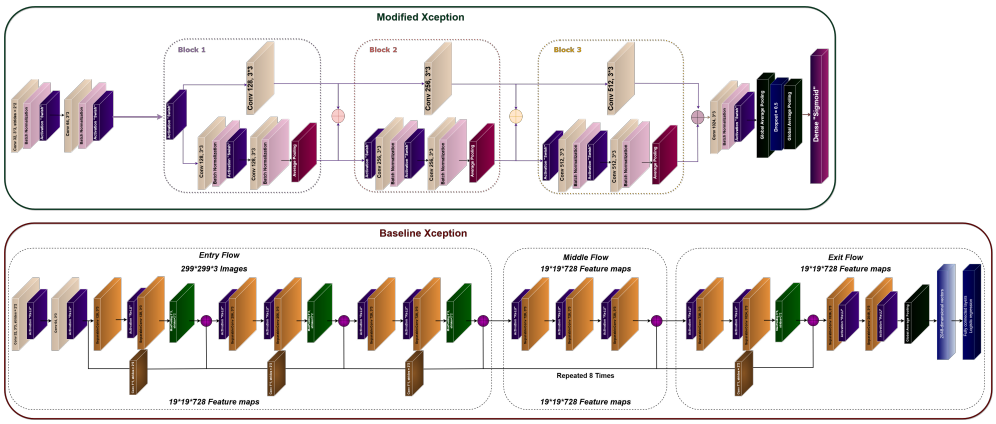

Fig. 14 Illustrates the layers of the baseline Xception model and the modified Xception model

Table 8 Ablation study on fusion methods on the CASIA-FASD and Replay-Attack datasets with metrics (EER (%) and HTER (%))

Fusion methods & Datasets	CASIA-FASD		Replay-Attack	
	EER	HTER	EER	HTER
Attention-mechanism (Our approach)	**0**	**0.05**	**0**	**0.01**
Concatenate (Antil et al. [45])	2.37	3.20	0	0
Weighted-average (Atoum et al. [41])	2.67	2.27	0.79	0.72

Table 9 Ablation study evaluation of the modified Xception model on the CASIA-FASD and Replay-Attack datasets with metrics (EER (%) and HTER (%))

Methods & Datasets	CASIA-FASD		Replay-Attack	
	EER	HTER	EER	HTER
Our approach (Modified Xception)	**0**	**0.05**	**0**	**0.01**
Baseline Xception [55]	50.1	50	36.73	29.03

5 Conclusion and future work

In this paper, we introduce an innovative approach for detecting face spoofing using a combination of color texture descriptors and a novel Convolutional Neural Network (CNN) architecture. Our proposed method is built upon a unique CNN architecture consisting of two parallel branches. The first branch is designed to work with a robust Local Phase Quantization (LPQ) invariant descriptor, which is derived from the fusion of color and texture information within the Hue, Saturation, Value (HSV) color space.

This approach allows us to accurately capture the reflective properties of the face. The combination of the HSV color space with LPQ has been widely acknowledged to significantly enhance performance in this context. On the other hand, the second branch of our CNN model takes an RGB image as input, effectively segregating chromatic (color-related) information from achromatic (brightness-related) information. This separation enables us to extract essential facial color features. Each CNN branch independently produces a feature vector representing the extracted information. These two resulting feature vectors are then concatenated using the attention mechanism based fusion method, forming an input vector for the subsequent Dense layer, which is responsible for distinguishing between live and spoofed faces. Our approach's strength lies in its efficiency in extracting vital features. It accomplishes this by focusing on relevant color patterns and structures while filtering out extraneous luminance variations. The concatenation of these two branches enhances the overall robustness of our method, making it highly resilient to various attacks. Our method is proficient at detecting 2D facial spoofing attacks, including those involving printed photos and replayed videos. We conducted a series of experiments on the CASIA-FASD and Replay-Attack datasets, showcasing the effectiveness and superior performance of our approach compared to other state-of-the-art methods. Our results exhibit an ACER of 0.04%, APCER of 0.03%, FNR of 0.06%, EER of 0%, HTER of 0.05%, Accuracy of 96.00%, Precision of 97.00%, Recall of 97.00%, and F1-Score of 96.00% for the CASIA-FASD dataset. Similarly, for the Replay-Attack dataset, our approach demonstrates an ACER of 0.05%, APCER of 0%, FNR of 0.11%, EER of 0%, HTER of 0.01%, Accuracy of 97.00%, Precision of 97.00%, Recall of 98.00%, and F1-Score of 97.00%. Our results are indeed promising, showcasing the potential for improved face spoofing detection in real-world scenarios.

For future work, we plan to explore new architectures for deep learning and transfer learning methods using different combinations of databases. We also plan to exploit methods based on 3D depth information within the CNN model for face anti-spoofing.

Acknowledgments. This work received support from the General Directorate for Scientific Research and Technological Development within the Ministry of Higher Education and Scientific Research (DGRSDT) in Algeria.

Funding. This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

Data Availability. We have not utilized any proprietary data and we have provided comprehensive references for the publicly accessible datasets discussed in our paper.

Declarations

Conflict of interests. The authors declare that they have no conflict of interest.

References

[1] Anjos A, Marcel S (2011) Counter-measures to photo attacks in face recognition: a public database and a baseline. In 2011 international joint conference on Biometrics (IJCB). IEEE, pages 1–7. https://doi.org/10.1109/IJCB.2011.6117503

[2] Galbally J, Marcel S, Fierrez J (2014) Biometric antispoofing methods: A survey in face recognition. IEEE Access, 2:1530-1552. https://doi.org/10.1109/ACCESS.2014.2381273

[3] Hadid A, Evans N, Marcel S, Fierrez J (2015) Biometrics systems under spoofing attack: An evaluation methodology and lessons learned. IEEE Signal Processing Magazine, 32(5):20–30. https://doi.org/10.1109/MSP.2015.2437652

[4] Li Y, Xu K, Yan Q, Li Y, Deng RH (2014) Understanding OSN-based facial disclosure against face authentication systems. In Proceedings of the 9th ACM symposium on Information, computer and communications security, pages 413-424. https://doi.org/10.1145/2590296.2590315

[5] Zhang Z, Yan J, Liu S, Lei Z, Yi D, Li SZ (2012) A face antispoofing database with diverse attacks. In 2012 5th IAPR international conference on Biometrics (ICB). IEEE, pages 26-31. https://doi.org/10.1109/ICB.2012.6199754

[6] Cortes C, Vapnik V (1995) Support-vector networks. Machine learning. Springer, 20:273-297. https://doi.org/10.1007/BF00994018

[7] Tharwat A, Gaber T, Ibrahim A, Hassanien AE (2017) Linear discriminant analysis: A detailed tutorial. AI communications. IOS Press, 30(2):169–190. https://doi.org/10.3233/AIC-170729

[8] Määttä J, Hadid A, Pietikäinen M (2011) Face spoofing detection from single images using micro-texture analysis. IEEE international joint conference on Biometrics (IJCB), pp. 1–7. https://doi.org/10.1109/IJCB.2011.6117510

[9] Yang J, Lei Z, Liao S and Li SZ (2013) Face liveness detection with component dependent descriptor. International Conference on Biometrics (ICB), pp 1—6. https://doi.org/10.1109/ICB.2013.6612955

[10] Dalal N, Triggs B (2005) Histograms of oriented gradients for human detection. IEEE computer society conference on computer vision and pattern recognition (CVPR'05), vol. 1, pp. 886—893. https://doi.org/10.1109/CVPR.2005.177

[11] Ojansivu V, Heikkilä J (2008) Blur insensitive texture classification using local phase quantization. In International conference on image and signal processing. Springer, pages 236–243. https://doi.org/10.1007/978-3-540-69905-7_27

[12] Lazebnik S, Schmid C, Ponce J (2006) Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. IEEE computer society

conference on computer vision and pattern recognition (CVPR'06), vol. 2, pp. 2169—2178. https://doi.org/10.1109/CVPR.2006.68

[13] Chingovska I, Anjos A, Marcel S (2012) 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, pages 1–7.

[14] Boulkenafet Z, Komulainen J, Hadid A (2015) Face anti-spoofing based on color texture analysis. IEEE international conference on image processing (ICIP), pp. 2636—2640. https://doi.org/10.1109/ICIP.2015.7351280

[15] Boulkenafet Z, Komulainen J, Hadid A (2016) Face spoofing detection using colour texture analysis. IEEE Transactions on Information Forensics and Security, 11(8):1818–1830. https://doi.org/10.1109/TIFS.2016.2555286

[16] Boulkenafet Z, Komulainen J, Hadid A (2016) Face antispoofing using speeded-up robust features and fisher vector encoding. IEEE Signal Processing Letters, 24(2), pp. 141—145. https://doi.org/10.1109/LSP.2016.2630740

[17] Wen D, Han H, Jain AK (2015) Face spoof detection with image distortion analysis. IEEE Transactions on Information Forensics and Security, 10(4), pp. 746–761. https://doi.org/10.1109/TIFS.2015.2400395

[18] Singh AK, Joshi P, Nandi GC (2014) Face recognition with liveness detection using eye and mouth movement. IEEE international conference on signal propagation and computer technology (ICSPCT 2014), pp. 592—597. https://doi.org/10.1109/ICSPCT.2014.6884911

[19] Jain A, Nandakumar K, Ross A (2005) Score normalization in multimodal biometric systems. Elsevier Pattern recognition, 38(12), pp. 2270—2285. https://doi.org/10.1016/j.patcog.2005.01.012

[20] George A, Marcel S (2019) Deep pixel-wise binary supervision for face presentation attack detection. In 2019 International Conference on Biometrics (ICB). IEEE, pages 1–8. https://doi.org/10.1109/ICB45273.2019.8987370

[21] Huang G, Liu Z, Van Der Maaten L, Weinberger KQ (2017) Densely connected convolutional networks. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 4700–4708. https://doi.org/10.48550/arXiv.1608.06993

[22] Abdullakutty F, Johnston P, Elyan E (2022) Fusion Methods for Face Presentation Attack Detection. Sensors. MDPI, 22(14):5196. https://doi.org/10.3390/s22145196

[23] Abdullakutty F, Elyan E, Johnston P, Ali-Gombe A (2022) Deep transfer learning on the aggregated dataset for face presentation attack detection. Cognitive

computation. 14(6):2223–2233. https://doi.org/10.1007/s12559-022-10037-z

[24] Satapathy A, Livingston LM, Jenila (2021) A lite convolutional neural network built on permuted Xceptio-inception and Xceptio-reduction modules for texture based facial liveness recognition. Multimedia Tools and Applications. Springer, 80:10441–10472. https://doi.org/10.1007/s11042-020-10181-4

[25] Gwyn T, Roy K (2022) Examining gender bias of convolutional neural networks via facial recognition. Future Internet, Multidisciplinary Digital Publishing Institute, 14(12):375. https://doi.org/10.3390/fi14120375

[26] Wang D, Ma G, Liu X (2022) An intelligent recognition framework of access control system with anti-spoofing function. AIMS Mathematics, 7(6):10495–10512. https://doi.org/10.3934/math.2022585

[27] Li L, Feng X, Boulkenafet Z, Xia Z, Li M, Hadid A (2016) An original face anti-spoofing approach using partial convolutional neural network. In 2016 Sixth International Conference on Image Processing Theory, Tools and Applications (IPTA). IEEE, pages 1–6. https://doi.org/10.1109/IPTA.2016.7821013

[28] Yang X, Luo W, Bao L, Gao Y, Gong D, Zheng S, Li Z, Liu W (2019) Face anti-spoofing: Model matters, so does data. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 3507–3516. https://doi.org/10.1109/CVPR.2019.00362

[29] Deb D, Jain AK (2020) Look locally infer globally: A generalizable face anti-spoofing approach. IEEE Transactions on Information Forensics and Security, 16:1143–1157. https://doi.org/10.1109/TIFS.2020.3029879

[30] Shao R, Lan X, Li J, Yuen PC (2019) Multi-adversarial discriminative deep domain generalization for face presentation attack detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 10023–10031. https://doi.org/10.1109/CVPR.2019.01026

[31] de Souza GB, Papa JP, Marana AN (2018) On the learning of deep local features for robust face spoofing detection. In 2018 31st SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI). IEEE, pages 258–265. https://doi.org/10.1109/SIBGRAPI.2018.00040

[32] Sun CY, Chen SL, Li XJ, Chen F, Yin XC (2022) Danet: Dynamic attention to spoof patterns for face anti-spoofing. In 2022 26th International Conference on Pattern Recognition (ICPR). IEEE, pages 1929–1936. https://doi.org/10.1109/ICPR56361.2022.9956725

[33] Kong Y, Li X, Hao G, Liu C (2022) Face Anti-Spoofing Method Based on Residual Network with Channel Attention Mechanism. Journal of Electronics. MDPI, 11(19):3056. https://doi.org/10.3390/electronics11193056

[34] Liu Y, Jourabloo A, Liu X (2018) Learning deep models for face anti-spoofing: Binary or auxiliary supervision. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 389–398. https://doi.org/10.48550/arXiv.1803.11097

[35] da Silva VL, Lérida JL, Sarret M, Valls M, Giné F (2023) Residual spatiotemporal convolutional networks for face anti-spoofing. Journal of Visual Communication and Image Representation. Elsevier, page 103744. https://doi.org/10.1016/j.jvcir.2022.103744

[36] Xu Z, Li S, Deng W (2015) Learning temporal features using LSTM-CNN architecture for face anti-spoofing. In 2015 3rd IAPR asian conference on pattern recognition (ACPR). IEEE, pages 141–145. https://doi.org/10.1109/ACPR.2015.7486482

[37] Guo J, Zhu X, Xiao J, Lei Z, Wan G, Li SZ (2019) Improving face anti-spoofing by 3d virtual synthesis. In 2019 International Conference on Biometrics (ICB). IEEE, pages 1–8. https://doi.org/10.1109/ICB45273.2019.8987415

[38] Hashemifard S, Akbari M (2021) A compact deep learning model for face spoofing detection. arXiv preprint arXiv:2101.04756. https://doi.org/10.48550/arXiv.2101.04756

[39] Khammari M (2019) Robust face anti-spoofing using CNN with LBP and WLD. IET Image Processing, Wiley Online Library, 13(11):1880–1884. https://doi.org/10.1049/iet-ipr.2018.5560

[40] Patel K, Han H, Jain AK (2016) Cross-database face antispoofing with robust feature representation. Biometric Recognition: 11th Chinese Conference, CCBR 2016, Chengdu, China, October 14-16, 2016, Proceedings 11. Springer, 611–619. https://doi.org/10.1007/978-3-319-46654-5_67

[41] Atoum Y, Liu Y, Jourabloo A, Liu X (2017) Face anti-spoofing using patch and depth-based CNNs. In 2017 IEEE International Joint Conference on Biometrics (IJCB). IEEE, pages 319–328. https://doi.org/10.1109/BTAS.2017.8272713

[42] Chen H, Hu G, Lei Z, Chen Y, Robertson NM, Li SZ (2019) Attention-based two-stream convolutional networks for face spoofing detection. IEEE Transactions on Information Forensics and Security, 15:578–593. https://doi.org/10.1109/TIFS.2019.2922241

[43] Wang Y, Nian F, Li T, Meng Z, Wang K (2017) Robust face anti-spoofing with depth information. Journal of Visual Communication and Image Representation. Elsevier, 49:332–337. https://doi.org/10.1016/j.jvcir.2017.09.002

[44] Asim M, Ming Z, Javed MY (2017) CNN based spatio-temporal feature extraction for face anti-spoofing. In 2017 2nd International Conference on Image, Vision and

Computing (ICIVC). IEEE, pages 234–238. https://doi.org/10.1109/ICIVC.2017.7984552

[45] Antil A, Dhiman C (2023) A two stream face anti-spoofing framework using multi-level deep features and ELBP features. Multimedia Systems. Springer, pages 1–16. https://doi.org/10.1007/s00530-023-01060-7

[46] Feng L, Po LM, Li Y, Xu X, Yuan F, Cheung TCH, Cheung KW (2016) Integration of image quality and motion cues for face anti-spoofing: A neural network approach. Journal of Visual Communication and Image Representation. Elsevier, 38:451–460. https://doi.org/10.1016/j.jvcir.2016.03.019

[47] Shu X, Li X, Zuo X, Xu D, Shi J (2023) Face spoofing detection based on multi-scale color inversion dual-stream convolutional neural network. Expert Systems with Applications. Elsevier, 224:119988. https://doi.org/10.1016/j.eswa.2023.119988

[48] Zhang K, Zhang Z, Li Z, Qiao Y (2016) Joint face detection and alignment using multitask cascaded convolutional networks. IEEE signal processing letters, 23(10):1499–1503. https://doi.org/10.1109/LSP.2016.2603342

[49] Bargshady G, Zhou X, Deo RC, Soar J, Whittaker F, Wang H (2020) The modeling of human facial pain intensity based on temporal convolutional networks trained with video frames in HSV color space. Applied Soft Computing. Elsevier, 97:106805. https://doi.org/10.1016/j.asoc.2020.106805

[50] Rahman MA, Purnama IKE, Purnomo MH (2014) Simple method of human skin detection using HSV and YCbCr color spaces. In 2014 International Conference on Intelligent Autonomous Agents, Networks and Systems. IEEE, pages 58–61. https://doi.org/10.1109/INAGENTSYS.2014.7005726

[51] Xiao Y, Cao Z, Wang L, Li T (2017) Local phase quantization plus: A principled method for embedding local phase quantization into fisher vector for blurred image recognition. Information Sciences. Elsevier, 420:77–95. https://doi.org/10.1016/j.ins.2017.08.059

[52] Ramachandran P, Zoph B, Le QV (2017) Searching for activation functions. arXiv preprint arXiv:1710.05941. Technical report, 7(1):5. https://doi.org/10.48550/arXiv.1710.05941

[53] Nair V, Hinton GE (2010) Rectified linear units improve restricted boltzmann machines. In Proceedings of the 27th international conference on machine learning (ICML-10), pages 807–814.

[54] Iliev A, Kyurkchiev N, Markov S (2017) On the approximation of the step function by some sigmoid functions. Mathematics and Computers in Simulation. Elsevier, 133:223–234. https://doi.org/10.1016/j.matcom.2015.11.005

[55] Chollet F (2017) Xception: Deep learning with depthwise separable convolutions. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 1251–1258. https://doi.org/10.48550/arXiv.1610.02357

[56] Lanjewar MG, Morajkar P, P P (2023) Modified transfer learning frameworks to identify potato leaf diseases. Multimedia Tools and Applications. Springer, pages 1–23. https://doi.org/10.1007/s11042-023-17610-0

[57] Lanjewar MG, Gurav OL (2022) Convolutional Neural Networks based classifications of soil images. Multimedia Tools and Applications. Springer, 81(7):10313–10336. https://doi.org/10.1007/s11042-022-12200-y

[58] Abadi M, Agarwal A, Barham P, Brevdo E, Chen Z, Citro C, Corrado GS, Davis A, Dean J, Devin M, et al. (2016) Tensorflow: Large-scale machine learning on heterogeneous distributed systems. arXiv preprint arXiv:1603.04467. https://doi.org/10.48550/arXiv.1603.04467

[59] Ho Y, Wookey S (2019) The real-world-weight cross-entropy loss function: Modeling the costs of mislabeling. IEEE access, 8:4806–4813. https://doi.org/10.1109/ACCESS.2019.2962617

[60] Kingma DP, Ba J (2014) Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980. https://doi.org/10.48550/arXiv.1412.6980