TLA Framework—A Transfer Learning Based Approach for Face Anti-spoofing

H. Vinutha* and G. Thippeswamy

***Abstract*—Security-based applications seek face biometrics as an integral part of the biometrics system which is susceptible to spoof attacks. A malicious person can gain unauthorized access to any system by displaying a picture or video of the registered user's face. Anti-spoofing techniques are becoming increasingly crucial in the face biometric authentication systems. Convolution Neural Networks (CNN) have recently gained popularity in important computer vision application areas, encouraging their usage for face spoof detection. Even though deep networks are more resistant, such designs require expensive computational training. Also, the adoption of deep CNN architectures for face anti-spoofing applications has been constrained by the lack of sufficient training data that the existing spoof datasets can offer. Also trained models to lack generalizability concerning unknown data domains, and are not robust enough to handle unseen attacks. We propose a Transfer Learnt Anti-spoof (TLA) framework in this paper, to induce and improve generalizability and accuracy in spoof detection. TLA framework consists of two Convolution neural networks namely ResNet-34 and MobileNetV2. Here we pre-train these two CNN models on a larger dataset at the base. Then a dense classification layer is formed to classify the features obtained from the previous convolutional base, into the real and spoofed faces. The TLA Framework was applied efficiently over the NUAA Photo Imposter dataset and the models within the framework exhibited the highest accuracy of 99.76% and 99.60% respectively for spoof detection and tests demonstrate that TLA outperforms the state-of-the-art techniques.**

***Index Terms*—Transfer learnt anti-spoof framework, transfer learning, ResNet-34, MobileNetV2, face anti-spoofing**

I. Introduction

The advancement in using computer vision technology has gained a lot of attention concerning the usage of facial biometrics systems for personal authentication. Though the face biometric systems are achieving greater accuracies, they remain vulnerable to presentation attacks. The face biometric system is fooled by presenting a few artifacts like a video or a printed photograph against the sensor of any input device like a camera. Face anti-spoofing is indeterminate in protecting the facial biometrics system against such malicious acts. There are a variety of presentation/spoof attacks with which 2D photo replay and print attacks are the most common ones. Therefore, to prevent such spoofing attempts, we have many anti-spoofing algorithms in place to evaluate whether the incoming image or video comprises the real face or fake face. Face presentation attack detection/spoof detection is an arduous task for all face biometric systems.

Initial presentation attack detection methods relied on a few established methods, such as Eigenfaces, Local Binary Patterns (LBP), Histogram of Oriented Gradients (HOG), Discrete Cosine Transform (DCT), Discrete Wavelet Transform (DWT), and Speeded-Up Robust Features (SURF), which may be intended to address general image recognition issues but may not be sufficient to address the Face Anti-spoofing System (FAS) issue. Additionally, these hand-crafted methods rely on raw data, which is insufficient, especially in complex circumstances. Convolutional and deep neural networks have become increasingly popular for detecting spoofed faces in recent years [1]. Though Deep Convolutional Neural Networks (CNNs) have achieved promising success in detecting spoofed faces, they are data-hungry. They require a huge voluminous dataset to train before making any reasonable prediction, to avoid over-fitting and these models are computationally expensive too. To address this issue, the most recent technique used is transfer learning, in which CNNs train on larger datasets, which are from the same or different domains. Finally, a known network or classifier is improved using the images of a smaller database.

The generalizability of the learned feature space is a crucial factor for assessing a FAS system's performance. The trained model could perform exceptionally well in analyses within a dataset but fall short on unknown distribution. Various security risks are raised by this lack of generalization because not all attacks carried out by imposters are identified. Face anti-spoofing transfer the expertise for the same task from one domain with more data to another with less data. Transfer learning has two subcategories: domain adaptation and knowledge distillation [2]. Knowledge distillation adopts a teacher-student structure to create generalized feature space between source and destination domains rather than domain adaptation, which uses similarity metrics or adversarial learning. The goal of Domain Adaptation (DA) is to close the gap between the source and target domains while boosting the performance of the model on unknown input by using adversarial learning or similarity metrics. This method can be used when training data and test data do not correctly have the same distribution, which happens rather frequently in real-world applications and degrades performance. These transfer learning techniques help us to save on computational resources and achieve more accuracy with smaller datasets.

Our Face Anti-spoofing dataset—NUAA Photo Imposter dataset [3] is a publicly available one having 12,614 images, which is relatively a smaller dataset to be trained with deep learning techniques and thus can incur an over-fitting problem. Also training all the layers of a convolutional neural

Manuscript received October 28, 2022; revised December 14, 2022; accepted May 4, 2023.

H. Vinutha is with Department of Computer Science and Engineering, BMS Institute of Technology and Management, Bengaluru, India and Department of Computer Science and Engineering, Dr. Ambedkar Institute of Technology, Bengaluru, India.

G. Thippeswamy is with Department of Computer Science and Engineering, B M S Institute of Technology and Management, Bengaluru, India.

*Correspondence: vinuthah@gmail.com (H.V.)

network from the scratch, i.e., from the initial layer is expensive concerning both resources and time. Transfer learning [4] technique solves the above problems by reusing a previously learned model over a large dataset over a new problem or a new dataset.

In recent days, transfer learning has become an effective method to reuse an eminent pre-trained CNN model to take care of different machine learning problems. However, the utilization of transfer learned, the pre-trained deep network is a newer concept concerning liveness detection [5] and not many have explored it. Consequently, in the interest of looking into the face spoof detection problem, our main contribution is two transfer-learned, pre-trained deep models, ResNet34 and MobileNetV2 models. Both models give better accuracy by availing lesser training time and providing generalization over trained and tested datasets.

Our work investigates the transfer learning-based procedures of two CNN architectures for face spoof detection. The remaining part of this paper consists of the literature survey of the existing methods in Section II, and the proposed transfer learned anti-spoof framework in Section III, to assess the suggested strategies in Section IV, several experiments have been carried out and Section V summarizes the study's conclusions and research findings.

II. Related Works

Previous works on anti-spoofing in the literature, have followed many approaches. Major ones are texture-based, temporal-based, and frequency based. All the above approaches incorporated handcrafted texture features like LBP, HOG, and its variants followed by conventional classifiers like K-means, Support Vector Machines (SVM), or Neural networks to carry out the anti-spoofing task [6,7]. To distinguish between real faces and fraudulent ones, the temporal-based approaches [8] utilize facial motion patterns like eyes blinking or facial movements that use optical flow for face movement tracking. Some techniques [9] call for specialized 3D technologies that derive depth information from 2D photos, or the 3D shape information captured by 3D sensors is examined and contrasted with a real face. In a few techniques pulse signals are extracted from face images without making any contact with skin, such a method is based on Remote Photo Plethysmography (RPPG) [10]. Masks, fake face attacks, and assistance of auxiliary data make these systems vulnerable and also for above systems require depth information. Spoof detection can also be categorized based on spatial and temporal properties: static and dynamic features methods [11]. Dynamic procedures examine the images based on their temporal qualities while static approaches assess the images based on the spatial relationship in the image. Static techniques that utilize local binary patterns and their variants [6, 12, 13], Fourier analysis [7, 14], Difference-of-Gaussian (DoG) [14], and Lambertian models [15]. Dynamic methods benefit from the temporal correlation between subsequent frames of video [16]. In several works, facial expressions serve as a sign of an impending attack (e.g.: eye blink [17], face natural movements [18], or movement of lips). Above mentioned methods mostly rely on hand-crafted features. Few approaches utilize CNN [19] to extract features from images, thus recognizing the attacks. For example, technique that uses an AlexNet [20] to extract the features, and classify them using Support Vector Machines (SVM).

In previous works that were carried out, machine learning-based approaches were predominantly used for spoof detection. But later Deep learning-based techniques pitched into this domain [21, 22] which are representation-learning methods with different representations utilized at different levels of the neurons in Deep Neural Networks (DNN) [23]. Here the representations at one level are transformed into a higher level using a non-linear module. Also, DNNs are fed with unprocessed input data, unlike the conventional algorithms, later the DNNs perceive the representations required for classification, detection, or recognition. Hence techniques using deep learning have proven to be commendable tasks while determining the spoofed faces. A multi-modal fusion approach was suggested by Xiaoguang *et al.* [24] for Face Anti-spoofing by maintaining a variety of visual modalities on their displayed face images. An anti-spoofing network was used by Yang *et al.* [25] to consider temporal and spatial information that is both global and local. Liu *et al*. [26] generated a large dataset with spoofing attacks of 13 different types and utilized a deep tree network for the recognition of these spoofing attacking types. These deep learning-based techniques generally work well with trained face spoofing data, but performance degrades when new attack types are developed. But as the layers in the DNN get deeper, it seeks more data for training at each level and the standard datasets in our problem domain fail to cater to those numbers. Thus, a technique known as Transfer learning evolved out of the regular CNNs where a Machine Learning model yields trained knowledge to drive another model for a particular task [26, 27]. There are two ways to perform transfer learning. In the first approach, the source model acts as a feature extractor, and the selected layer's output in the CNN serves as input for the aimed model. The second approach tweaks the source model partially or completely and its weights are retrained using backpropagation. Transfer learning prevents the over-fitting problem in a large network when the data is too small in size to train from scratch. Computational resources are also spared using transfer learning, since conventional machine learning approaches may take from days to weeks to train from the scratch. Our approach for spoof detection is formed on transfer learning different pre-trained CNNs [5, 28].

III. Proposed TLA Framework

Face spoof detection attempts to differentiate between real and spoofed images. Usually, the number of fake faces presides over the real ones due to the various forms of attacks such as spoofed images. Hence imbalanced training data will be exposed to the system. Various datasets are available for face spoof detection problems. One of them is the NUAA Photo imposter dataset, which has been publicly available, that we have considered evaluating.

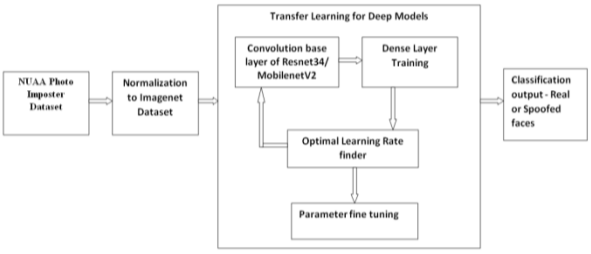


Fig. 1. Proposed face anti-spoofing framework with ResNet34/ MobileNetV2 deep models using Transfer learning.

The proposed Transfer Learnt Anti-spoof (TLA) framework (refer to Fig. 1) evaluates two CNN architectures, the first one being ResNet34 and the other being MobileNetV2 architectures. These two models were pre-trained using ImageNet [29], a larger classification dataset. From this pre-trained base model, we fine tune our ResNet34 and MobileNetV2 to improve generalization.

A. ResNet34Architecture

Deep learning has advanced significantly as a result of rapid technological advancements in hardware performance and computer technology [22]. Owing to their higher ability to categorize and recognize images, artificial neural networks have found widespread use in diverse areas [23]. CNN is a sophisticated, multilayer, fault-tolerant neural network. It also can learn on its own. It can solve issues in challenging circumstances with hazy backgrounds. Compared to other approaches, it has a far superior ability to generalize. An input layer, multiple convolutions, a few pooling layers, and completely linked and output layers are the typical layers that makeup CNN. It is used in natural language processing, computer vision, and other domains both unsupervised and supervised learning.

ResNet34 [26], which is described in "Deep Residual Learning for Image Recognition" is a modern image classification model, which is built as a 34-layer convolutional neural network. The ResNet-34 network's infrastructure is made up primarily of residual building blocks, which make up the entire network. The issue of the disappearance of gradient or explosion of gradient caused by increasing the neural network depth was effectively fixed by the residual construction block, which skipped the convolutional layers via a shortcut link. This improved the recognition rate for face spoof detection and gave more flexibility when building CNN structures. In the general context, Transfer is a carryover of skill or knowledge from one scenario to other.

In transition learning, a machine can exploit the expertise gained from the previous work to enhance the generalization of another. In normal neural networks, edges are detected in the initial layer, it gets shaped in layers in the center and the later layers carry some functional features.

Fig. 2 depicts the basic-organizational block's structure. The ResNet's 34 layers make use of the residual building component, which consists of several "Convolutions (Conv)", "Batch Normalizations (BN)", a "Rectified Linear Unit (ReLU)" activation function, and a short-cut connection. The following might be written as the result of the last construction block:

$$y = F(x) + x(1) \tag{1}$$

Here F is the residual function, x and y are the function input value and output value, respectively.

The first convolutional layer and some fundamental blocks make up the complete residual network.

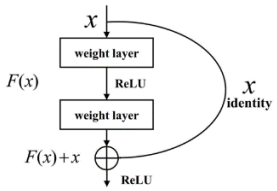


Fig. 2. ResNet 34 basic building block.

The ResNet-34 consists of a fully connected layer, an average pool layer, and a max-pooling layer with a size of 3×3. A traditional ResNet-34 model with 63.5 million parameters uses Rectifier Nonlinearity (ReLU) activation, BN, and the SoftMax function on the back of each convolution layer. In ANN, ReLU is a widely used activation function. We designated the regularized ReLU function as follows:

$$f(x) = \max(0, x) \tag{2}$$

Here f is the ReLU function and x is the input data.

The gradients of the sigmoid and tanh functions for deep networks are almost nil in the saturation zone. The gradients may readily disappear, which slows convergence and results in information loss. The ReLU gradient is often constant, which aids in resolving deep network convergence issues. ReLU, on the other hand, is closer in keeping with the traits of biological neurons because it has a unilateral function. Throughout the training process, the cross entropy loss function was used to update b. The following is a definition of the cross-entropy function:

$$H(p,q) = -n \sum_{i=1}^{i=n} p_i(x)\ \log_2 q_i\,(x) \tag{3}$$

For the input x, the probability is denoted by p, likelihood is denoted by q, and finally, H denotes cross-entropy, then the result is a probability. When compared to the variance loss function, the problem of updating weights and bias is too slow and is resolved by this method. The update of weights and deviations is also impacted by errors. Because of this, when the error is significant enough, weights update relatively quickly. Similarly, weights and deviations update slowly when the error is minor.

Overall architecture: Table I depicts the proposed TLA ResNet34's structure. The first convolutional layer has 64 filters and a 7×7 kernel, followed by a max-pooling layer. The stride is set to 2 in both instances. Conv2_x comprises the pooling layer and subsequent convolution layers. The way the residuals are coupled causes these layers to typically be grouped in pairs. The next two layers comprise the layers between the pool,/2, and the filter 128 ones, which have a kernel size of 3×3 and 64 filters, which are all repeated three times. These 2 layers have a kernel size of 3×3, filters count of 128, and are repeated four times in total. This continues up until the SoftMax and average pooling functions.

TABLE I: SIZES OF OUTPUTS AND CONVOLUTIONAL KERNELS FOR RESNET34

Layer Name	Output size	34-Layer
Conv1	112×112	7×7, 64, stride 2
Conv2_x	56×56	3×3 max pool, stride 2 $\begin{bmatrix} 3\times3, & 64 \\ 3\times3, & 64 \end{bmatrix} \times 3$
Conv3_x	28×28	$\begin{bmatrix} 3\times3, & 128 \\ 3\times3, & 128 \end{bmatrix} \times 4$
Conv4_x	14×4	$\begin{bmatrix} 3\times3, & 256 \\ 3\times3, & 256 \end{bmatrix} \times 6$
Conv5_x	7×7	$\begin{bmatrix} 3\times3, & 512 \\ 3\times3, & 512 \end{bmatrix} \times 3$
	1×1	Average pool, 1000-dc SoftMax

Refer to Fig. 3 for further understanding. Here ResNet architecture is implemented in 5 blocks. The first block contains 64 filters with a stride of 2, followed by max-pooling with a stride of 2. The architecture employs padding of 3, since there is a chance of internal covariate shift. So, the network must be stabilized through BN. Finally, ReLU is used.

B. *Transfer Learnt Restnet34*

Several annotated datasets are required to train the CNN in order to deliver an excellent performance [4]. However, gathering such a massive amount of data is difficult, and classifying the images becomes expensive. Transfer learning, which has been demonstrated to be a very successful method, is therefore utilized to train the neural network, whenever small datasets are involved. The little amount of data in this experiment makes it simple to overfit problems, and the model also needs more training epochs, which reduces the model's capacity to recognize patterns.

Thus, pre-training the model on ImageNet using transfer learning can enhance the classification of real or fake faces. It took less time to train ResNet-34 because it was adjusted to accommodate the data in this article.

The ImageNet dataset, which comprises more than 100,000 images in 200 different classes, serves as the pre-training data for Restnet34. RestNet distinguishes itself from traditional neural networks by using the residuals from each layer in the connected layers. Starting from the foundation state, we will fine-tune our ResNet model starting from the pre-trained checkpoint. This method is also frequently referred to as "transfer learning".

In the general context, transfer is a carryover of skill or knowledge from one scenario to other. In transition learning, a machine can exploit the expertise gained from the previous work to enhance the generalization of another. In normal neural networks, edges are detected in the initial layer, it gets shaped in layers in the center and the later layers carry some functional features.

In transfer learning, we make use of the initial and middle layers and later layers are replaced. Pre-trained models reduce the time required for feature engineering and training drastically. So, we use the model which has a large dataset to make the best of the training. The custom ResNet34 architecture employed here has been trained in advance on the ImageNet dataset. Having a well-trained model in hand, the model is started from the pre-trained checkpoint and ResNet34 is fine-tuned from this base state.

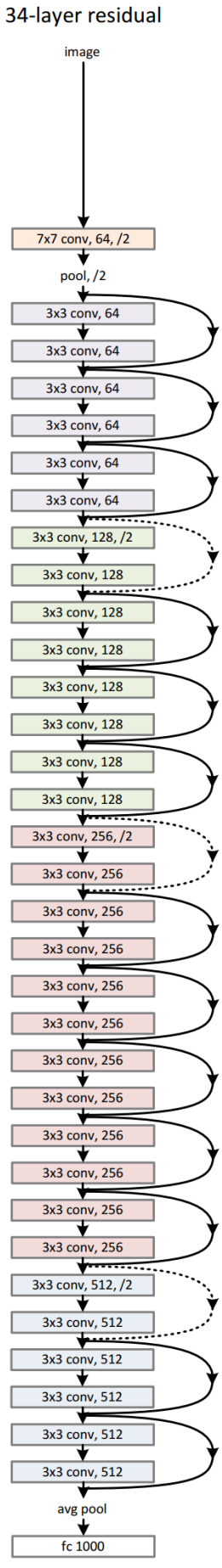


Fig. 3. ResNet 34 architecture

When loaded into the fastai data loader, the NUAA dataset is normalized to the standard deviation and mean of the ImageNet dataset. The pre-trained ResNet model from the torch vision model library is used in the following phase. We've initialized our model, fine-tuned the final layer, and frozen the remaining portions of the model. The procedure described above teaches us the pertinent pre-trained traits. Even then the validation loss has not decreased for 20 epochs, then an early stopping callback is used to stop the training session. Next, the model's parameters are unfrozen, and the ideal learning rate is determined. The model won't learn much if the learning rate is too low, and we have to backpropagate the way off the map in function space if it is too high.

Next, we unfreeze the model and train it for 50 more epochs to fine-tune our model to yield maximum performance. As the performance increases the validation error rate decreases. In our case, its error rate is 0.002377, and the validation loss is 0.010181 which is quite small.

The NUAA dataset is trained quickly by freezing a layer to stop its weights from changing. It just trains a particular set of layers for NUAA rather than all the layers. Thus, by drawing on the knowledge and expertise of the bigger ImageNet dataset, improved accuracy can be attained even for smaller datasets like NUAA.

C. MobileNetV2 Architecture

MobileNet is a flexible and effective CNN architecture, that is used in various real-time applications to create lighter models by replacing standard convolutions with depth-wise separable convolutions. Model developers can choose to trade speed or accuracy using two global parameters, the width multiplier, and the resolution multiplier in MobileNet.

The core of MobileNet is a collection of depth-separable convolutional layers. Each depth-wise separable convolution layer is made up of a pointwise and a depth-wise convolution. A MobileNet has 28 layers if pointwise and depth-wise convolutions are counted as individual components. The 4.2 million parameters of a typical MobileNet can be further condensed by varying the width multiplier hyperparameter suitably. For the images in our dataset, these values measure 224×224×3.

MobileNet utilizes a global hyperparameter known as Width multiplier, "α" to build less complex and computationally feasible models. It has a value between 0 and 1. For a specific layer, the number of input channels "M" and the number of output channels "N" becomes $\alpha \times$ M and $\alpha \times$N, respectively. This reduces model size and computation costs. Both the number of parameters and the cost of computation are reduced by a factor of two. A few of the often-used values are 1, 0.75, 0.5, and 0.25. Another hyperparameter known as, Resolution multiplier, "ρ" is used to decrease the input image's resolution, which in turn decreases the input provided to each layer by the same amount. The resolution of the input image is $224 \times \rho$ for a particular value of ρ. As a result, the computational cost is decreased by a factor of 2.

MobileNetV2 architecture: Depth-wise Separable Convolution was added to MobileNetV1, which significantly reduced the model size and network's complexity cost, making it suitable for mobile devices with limited processing capacity. MobileNetV2 introduces an improved module with an inverted residual structure. This time, nonlinearities in thin layers are eliminated. When MobileNetV2 is used as the foundation for feature extraction, modern performance for object detection and semantic segmentation is achieved.

The MobileNetV1 architecture witnessed some important up gradation that improved the model's accuracy significantly. The use of the ReLU6 activation function in place of ReLU and the updating of linear bottlenecks and inverted residual blocks were the key variations made to the architecture. The MobileNetV2 architecture has the residual structure stacked up in the inverted form, where their input and output are thin bottleneck layers. It extracts the features in the expansion layer using lightweight convolutions. Then the non-linearities in the narrow layers are removed. Fig. 4 denotes the overall MobileNet V2 architecture. There are two different kinds of blocks in MobileNetV2. A single stride makes up the first block. A block that can shrink by two strides is an additional option. Every sort of block has three layers. ReLU6 serves as the initial layer of a 1×1 convolution. The second layer is depth-wise convolution.

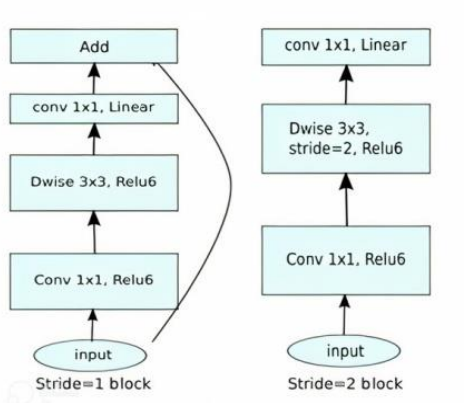


Fig. 4. MobileNet V2 architecture.

And another 1×1 convolution in the absence of non-linearity makes up the third layer (Refer to Table II). If ReLU is applied again, deep networks are said to have the power of a linear classifier only on the non-zero volume portion of the output domain. The t-expansion factor is another. $T = 6$ for all significant experiments. If the input had 64 channels the internal output would have $64 \times t = 64 \times 6 = 384$ channels.

TABLE II: THREE LAYERS IN EACH BLOCK OF MOBILENETV2

Input	Operator	Output
h × w × k	1 × 1 conv2d, ReLU6	h × w × (*t* k)
h × w ×*t* k	3 × 3 dwises=*s*, ReLU6	$\frac{h}{s} \times \frac{w}{s} \times$ (*t* k)
$\frac{h}{s} \times \frac{w}{s} \times t$ k	linear 1 × 1 conv2d	$\frac{h}{s} \times \frac{w}{s} \times$ k'

TABLE III: OVERALL ARCHITECTURE OF MOBILENETV2

Input	Operator	T	c	n	s
$224^2 \times 3$	Conv2d	-	32	1	2
$112^2 \times 32$	Bottleneck	1	16	1	1
$112^2 \times 16$	Bottleneck	6	24	2	2
$56^2 \times 24$	Bottleneck	6	32	3	2
$28^2 \times 32$	Bottleneck	6	64	4	2
$14^2 \times 64$	Bottleneck	6	96	3	1
$14^2 \times 96$	Bottleneck	6	160	3	2
$7^2 \times 160$	Bottleneck	6	320	1	1
$7^2 \times 320$	Conv2d 1 × 1	-	1280	1	1
$7^2 \times 1280$	avgpool 7 × 7	-	-	1	-
1 × 1 × 1280	Conv2d 1 × 1	-	k	-	

The overall design of MobileNetV2 is shown in Table III. Here, *n* is repeating number, *t* is expansion factor, and *s*is stride, and for spatial convolution, 33 kernels are employed.

D. Transfer Learnt MobileNetV2

For MobileNetV2 implementation, we use tensor flow. First, we convert our dataset into a Tensor flow Dataset using Image Folder API. All images of the NUAA dataset will be resized to 160×160. The base model is created using the MobileNetV2 model's pre-trained Covnents. The ImageNet dataset was used for the pre-training. The top classification layers are then removed from this pre-trained network, creating the perfect environment for feature extraction. Here, the final layer before the flatten operation, known as the "bottleneck layer," is taken into consideration rather than the final classification layer as in standard machine learning models. Compared to the previous layer, these layers' features are more generalizable. The previous step's convolutional basis is frozen and utilized as a feature extractor. The 160×160×3 image is converted into a 5×5×1280 block of features via the feature extractor. A classifier is added on top of it and trained. The convolutional base is frozen before the compilation and training of the model.

Freezing prevents the possible weight updating in a given layer during the training process. The features are converted into a 1280-element vector per image using Keras's GlobalAveragePooling2D layer. These features are then transformed into a final prediction via a dense layer, where positive integers are projected to be class 1 (a real face), and negative integers to be class 0 (a fake face). There is no requirement for an activation function in this scenario because the prediction will be treated as a logit or a raw prediction value.

IV. Experiments and Discussions

We evaluate the accuracy of the TLA framework for the face biometric system by conducting experiments on NUAA face anti-spoofing dataset. In the proposed framework, the weights were frozen from the third layer to the bottom layers and backpropagation was used to fine-tune them from the fourth block up to the initial layers. This technique was assessed using NUAA test folders. The implementation was carried out using fastai and pytorch and MobileNetV2 was implemented using tensor flow.

ResNet34 experimentation: On a single GPU, the suggested TL-ResNet34 was employed and trained.The model using the Stochastic Gradient Descent optimizer and Cross-entropy loss function is trained for 20 iterations with a batch size of 224 and a learning rate of 1×10^{-4}.

MobileNetV2 experimentation: In MobileNetV2, the base feature extractor layers, the global average layer, and prediction layers are stacked and the model is compiled again with RMSProp optimizer and Binary Cross entropy loss since there are two classes. This model was finetuned with a learning rate of 1×10^{-4}, batch size of 224, and 20 epochs.

The parameters used in both models are tabulated in Table IV and the performance of the two models of the TLA framework is evaluated in the experiments conducted.

TABLE IV: Parameters Used in the ResNet34 and MobileNetV2 Models

Parameter	ResNet34	MobileNetV2
Input shape	(32,32,3)	(32,32,3)
Weight	Initialized to ImageNet	Initialized to ImageNet
Optimizer	Stochastic Gradient Descent	RMSProp
Loss function	Binary Cross entropy	Binary Cross entropy
Classifier	SoftMax	SoftMax
Epochs	20	20
Batch size	224	224
Dropout rate	Nil	0.3
Regularization	BatchNormalization	BatchNormalization

A. Dataset

The first publicly accessible dataset for face anti-spoofing was the NUAA Photograph Imposter dataset [3]. These pictures were obtained in three sessions, with a two-week gap between each, using standard webcams in various settings with various lighting conditions. Attacks that are printed flat or warped are assessed. Sessions were created separately for training and testing. There are 15 subjects in this dataset. 500 photos are gathered for each topic, with 5105 real face images and 7509 fake ones. Each image features a frontal view of a face with an expressionless face. Fig. 5 shows some examples of it.

Fig. 5. Sample real (first five rows) and spoofed images (last five rows) from NUAA photo imposter dataset.

B. Evaluation Metrics

We have utilized the standardized ISO/IEC 30107-3 metrics [30] as the evaluation metrics for the Face anti-spoofing model. Mainly these metrics are akin to the types of errors and how they are measured and evaluated [31]. At the basic level, Spoof detection will incur False Positive (FP) and False Negative (FN) errors. Also, we evaluate two more errors: False Positive Rate (FPR), which is the ratio between false positive samples and the total number of negative samples, and False Negative Rate (FNR), which is the ratio between false negative samples and the total positive samples. Also, many authors have reported different error rates, but they are all equivalent. For example, the ratio of correctly classified positives is defined as True Positive Rate (TPR), which is computed as 1-FNR. The ratio of correctly detected negatives is defined as True Negative Rate (TNR), which is computed as 1-FPR. Similarly, a FPR can be termed a False Acceptance Rate (FAR) or Attack Presentation Classification Error Rate (APCER). All three metrics have different names given by different researchers but the calculations remain the same. FNR is also called a False Rejection Rate (FRR) or Normal Presentation Classification Error Rate (NPCER). Hence for measuring the error rate of spoofed or real faces, FPR and FNR are utilized. Table V lists the different Performance metrics that we have evaluated for our Deep models.

TABLE V: PERFORMANCE METRICS FOR FACE ANTI-SPOOFING

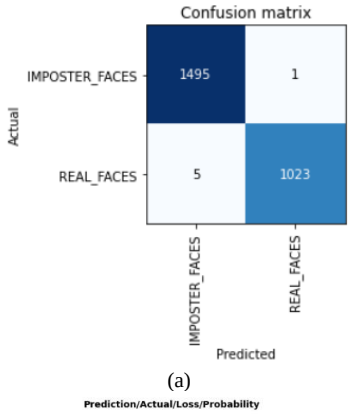
Performance Metric	Formula
False Positive Rate (FPR) OR False Acceptance Rate (FAR) OR Attack Presentation Classification Error Rate (APCER)	FPR or FAR or APCER = FP / (FP + TN)
False Negative Rate (FNR) OR False Rejection Rate (FRR) OR Normal Presentation Classification Error Rate (NPCER)	FNR or FRR or NPCER = FN / (FN + TP)
Half Total Error Rate (HTER) OR Average Classification Error Rate (ACER)	HTER or ACER= (FPR+FNR) / 2

ResNet34 Performance: We can evaluate our custom ResNet Image classification model's performance by using it for test inference and it is visualized in form of a Confusion matrix as shown in Fig. 6(a).

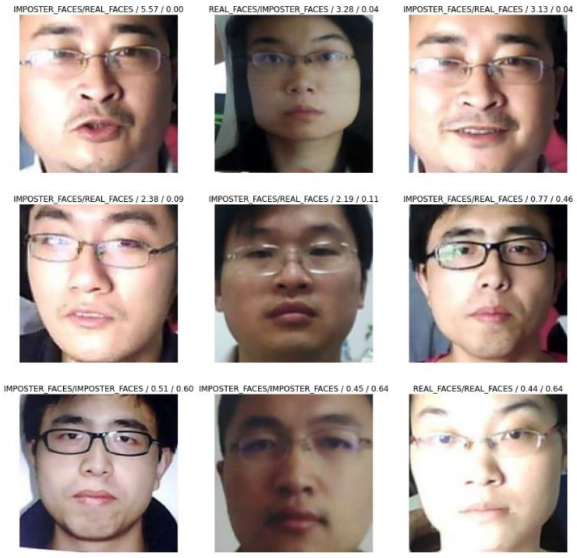
We run inference on our test set, images that our model has never seen. An efficient model was found with an Accuracy of 99.7623%, Precision of 0.9993, Recall of 0.9966, and validation loss value, as low as 0.0102 (Refer to Table VI). The time taken by our ResNet34 model to learn the parameters and train the network to predict the real and fake faces is 44 seconds. Also, our model's top losses are shown in Fig. 6(b), which plots the predicted label over the actual label what the loss incurred in each case, and its probability. Model evaluation was performed using FAR, FRR, and HTER. Lower FRR indicates that a lesser number of real faces are classified as spoofed faces. Also, a lower FAR indicates less number of spoofed faces is incorrectly identified as real faces. So, our transfer learned ResNet34 model exhibited FAR of 0.00097%, FRR of 0.00333%, and HTER of 0.00213%, whose values are very optimistic thereby increasing the efficacy of the face biometric system.

TABLE VI: RESULTS OF RESNET34

Technique	Dataset	Accuracy%	Precision	Recall	F1-Score
TLA-Res Net34	NUAA Photo Impostor	99.7623	0.9993	0.9966	0.9979



(a)



(b)

Fig. 6. (a) Classifier performance;(b) Top losses of our custom ResNet34 model.

MobileNetV2 Performance: In MobileNetV2, the base feature extractor layers, the global average layer, and the prediction layers are stacked and the model is compiled again with RMSProp optimizer and Binary Cross entropy loss since there are two classes. The initial loss and initial accuracy are evaluated before training our model, and our model's initial loss is 0.87 and initial accuracy is 0.51. After 20 epochs our custom MobileNetV2 model recurred a loss of 0.0222 and faired greatly with a performance accuracy of 99.55 %. The time taken by our MobileNetV2 model to learn the parameters and train the network to predict the real and fake faces is 197 seconds. We plot the loss and accuracy learning curves (see Fig. 7 to evaluate the model's performance during training and validation. The MobileNet model performed and converged very well. To improve its accuracy the model is fine-tuned. Here all the layers are frozen before the 100th layer. Restarting the training improved the accuracy and loss dramatically.

But the issue of negative transfer is one of transfer learning's main drawbacks. The proposed framework functions only when the initial and target issues are

sufficiently comparable for the initial training to be applicable. When the features learned by the bottom layer (the classification layer) are insufficient to distinguish the classes for the given problem set, transfer learning will not be useful. The features are poorly transferred when the datasets are not comparable.

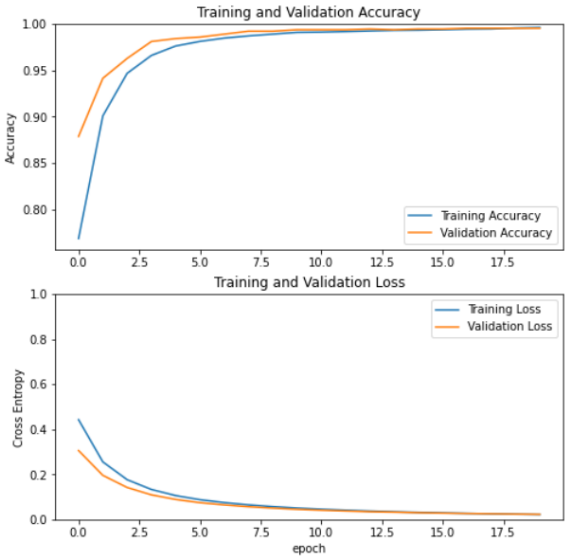


Fig. 7. Learning curves of the training and validation accuracy/loss for MobileNetV2.

Table VII lists the test accuracies of the different Transfer learned Deep networks for Face anti-spoofing, including the proposed models.

TABLE VII. TEST ACCURACIES OF DIFFERENT TRANSFER LEARNT DEEP MODELS

Technique	Attack Type	Accuracy%	HTER%
Transfer learnt VGG-16	Presentation (CASIA)	93.75	-
FasNet (Transfer learnt CNN-VGG16)	Presentation (REPLAY-ATTACK)	99.04	1.20
TLA-Renet34 (proposed)	Presentation (NUAA)	99.76	0.00213
TLA-MobileNet V2 (proposed)	Presentation (NUAA)	99.6	0.0024

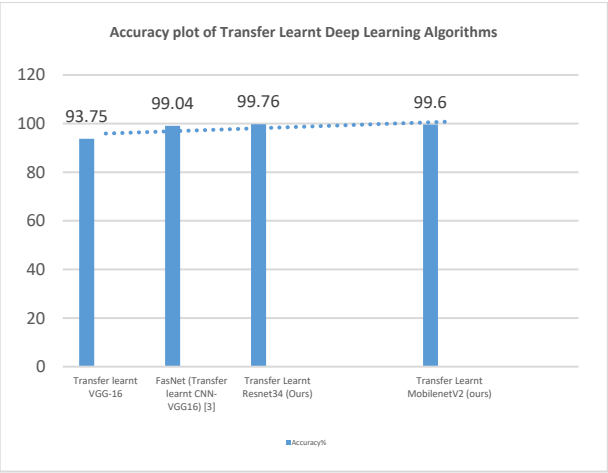


Fig. 8. Accuracy for proposed TLA smodels with SOTA.

Fig. 8 depicts the accuracy plots of various State-of-the-Art (SOTA) techniques with the models of our TLA framework. The results suggest that our transfer learned models outperformed the available transfer learned SOTA CNN models.

V. CONCLUSION

In this research work, we introduced the TLA framework with two deep models, ResNet34 and MobileNetV2 based on Transfer learning to detect spoofing attacks in the face biometric systems. We discussed how pre-trained architectures are modeled internally. We demonstrated how the base models helped in extracting features, and the dense layers classify the test set. In the end, we inspected how the feature transfer influenced the aimed problem from the pre-trained CNNs. The experimental results showed classification accuracies of 99.76% with HTER of 0.00213% and 99.6% with HTER of 0.0024% for transfer learned ResNet34 and MobileNetV2 models, respectively. Also, when transfer learned ResNet34 and MobileNetV2 were used on the test dataset, the fluctuation ranges of the loss curve and the accuracy curve were minimal. The HTER values denote that our models based on transfer learning increase the performance of the face anti-spoofing model.

The TLA Framework provides a generalized solution using a different dataset which is huge for feature extraction and training. This learning is transferred onto and used concerning smaller datasets for spoof detection. Using transfer learning in tandem with the model improves its performance.

But Transfer learning will not be effective if the features learned by the classification layer are insufficient to identify the classes for the given problem set. When the datasets are not comparable, the transfer of the features is poor.

As a direction for the future, we wish to experiment and propose Domain Agnostic models that handle the domain biases in the test datasets and also hope to create and detect the zero-shot spoof attacks multiplexed with available datasets to cover a larger spoof attack space.

CONFLICT OF INTEREST

The authors declare no conflict of interest.

AUTHOR CONTRIBUTIONS

H. Vinutha conducted the research and analyzed the data, designed the model and the computational framework, and also carried out the implementation. H. Vinutha wrote the manuscript with the support from G. T hippeswamy. G. Thippeswamy encouraged to investigate and supervised the findings of this work. All authors discussed the results and contributed to the final manuscript. All authors had approved the final version.

REFERENCES

[1] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[2] A. Sabaghi, M. Oghbaie, K. Hashemifard, and M. Akbari, "Deep learning meets liveness detection: Recent advancements and challenges," arXiv preprint, arXiv:2112.14796, 2021.

[3] X. Tan, Y. Li, J. Liu, and L. Jiang, "Face liveness detection from a single image with sparse low rank bilinear discriminative model," in *Proc. 11th European Conference on Computer Vision (ECCV'10)*, Crete, Greece, 2010.

[4] B. Jin, C. Leandro, and G. Nuno, "Deep facial diagnosis: Deep transfer learning from face recognition to facial diagnosis," *IEEE Access*, vol. 8, pp. 123649–123661,2020.

[5] A. J. Oeslle, R. S. Moia, E. Valle, and R. Lotufo, "Transfer learning using convolutional neural networks for face anti-spoofing," in *Proc. International Conference Image Analysis and Recognition*, Springer, Cham, 2017, pp. 27–34.
[6] G. Thippeswamy, H. Vinutha, and R. Dhanapal, "A new ensemble of texture descriptors based on local appearance-based methods for face anti-spoofing system," *Journal of Critical Reviews*, 2020. doi: 10.31838/jcr.07.11.118
[7] H, Vinutha and G. Thippeswamy, "Hand crafted face descriptor based on stockwell transform to detect 2d presentation attacks," *Neuro Quantology*, vol. 20, issue 10, 2022.
[8] Budiarto, "Face anti-spoofing method with blinking eye and HSV texture analysis," in *Materials Science and Engineering, IOP Conference Series*, vol. 1007, 012034, 2020. doi: 10.1088/1757-899X/1007/1/012034
[9] Y. Wang, F. Nian, T. Li, Z. Meng, and K. Wang, "Robust face anti-spoofing with depth information," *Journal of Visual Communication and Image Representation*, vol. 49, pp. 332–337,2017.
[10] J. Hernandez-Ortega, J. Fierrez, A. Morales, and P. Tome, "Time analysis of pulse-based face anti-spoofing in visible and NIR," in *Proc. the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2018, pp. 544–552.
[11] Z. Wang, Y. Xu, L. Wu, H. Han, Y. Ma, and G. Ma, "Multi-perspective features learning for face anti-spoofing," in *Proc. the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 4116–4122.
[12] I. Chingovska, A. Anjos, and S. Marcel, "On the effectiveness of local binary patterns in face anti-spoofing," in *Proc. the International Conference of Biometrics Special Interest Group (BIOSIG 2012)*, IEEE, 2012, pp. 1–7.
[13] M. Khammari, "Robust face anti-spoofing using CNN with LBP and WLD," *IET Image Processing*, vol. 13, no. 11, pp. 1880–1884, 2019.
[14] D. Yi, Z. Lei, Z. Zhang, and S. Z. Li, "Face anti-spoofing: Multi-spectral approach," in *Handbook of Biometric Anti-spoofing*, Springer, London, 2014, pp. 83–102.
[15] A. Anjos, J. Komulainen, S. Marcel, A. Hadid, and M. Pietikäinen, "Face anti-spoofing: Visual approach," in *Handbook of Biometric Anti-spoofing*, Springer, London, 2014, pp. 65–82.
[16] T. F. Pereira, J. Komulainen, A. Anjos, J. M. Martino, A. Hadid, M. Pietikäinen, and S. Marcel, "Face liveness detection using dynamic texture," *EURASIP Journal on Image and Video Processing*, vol. 2014, no. 1, pp. 1–15, 2014.
[17] K. Patel, H. Han, and A. K. Jain, "Cross-database face anti-spoofing with robust feature representation," in *Proc. Chinese Conference on Biometric Recognition*, Springer, Cham, 2016, pp. 611–619.
[18] X. Tu, H. Zhang, M. Xie, Y. Luo, Y. Zhang, and Z. Ma, "Enhance the motion cues for face anti-spoofing using CNN-LSTM architecture," arXiv preprint, arXiv:1901.05635, 2019.
[19] J. Gan, S. Li, Y. Zhai, and C. Liu, "3D convolutional neural network based on face anti-spoofing," in *Proc. 2017 2nd International Conference on Multimedia and Image Processing (ICMIP)*, IEEE, 2017, pp. 1–5.
[20] K. Ito, T. Okano, and T. Aoki, "Recent advances in biometrie security: A case study of liveness detection in face recognition," in *Proc. 2017 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, IEEE, 2017, pp. 220–227.
[21] Y. Liu, A. Jourabloo, and X. Liu, "Learning deep models for face anti-spoofing: Binary or auxiliary supervision," in *Proc. the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 389–398.
[22] H. Li, P. He, S. Wang, A. Rocha, X. Jiang, and A. C. Kot, "Learning generalized deep feature representation for face anti-spoofing," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 10, pp. 2639–2652, 2018.
[23] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Networks*, vol. 61, pp. 85–117, 2015.
[24] X. Tu, H. Zhang, M. Xie, Y. Luo, Y. Zhang, and Z. Ma, "Deep transfer across domains for face anti-spoofing," *Journal of Electronic Imaging*, vol. 28, no. 4, 043001, 2019.
[25] Q. Yang, X. Zhu, J.-K. Fwu, Y. Ye, G. You, and Y. Zhu, "PipeNet: Selective modal pipeline of fusion network for multi-modal face anti-spoofing," in *Proc. the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2020, pp. 644–645.
[26] Y. Liu, J. Stehouwer, A. Jourabloo, and X. Liu, "Deep tree learning for zero-shot face anti-spoofing," in *Proc. the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4680–4689.
[27] L. Torrey and J. Shavlik, "Transfer learning," in *Handbook of Research on Machine Learning Applications and Trends: Algorithms, Methods, and Techniques*, IGI global, 2010, pp. 242–264.
[28] S. Tammina, "Transfer learning using VGG-16 with deep convolutional neural network for classifying images," *International Journal of Scientific and Research Publications (IJSRP)*, vol. 9, no. 10, pp. 143–150, 2019.
[29] R. Quan, Y. Wu, X. Yu, and Y. Yang, "Progressive transfer learning for face anti-spoofing," *IEEE Transactions on Image Processing*, vol. 30, pp. 3946–3955, 2021.
[30] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang *et al*., "ImageNet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.
[31] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.