EURASIP Journal on Audio, Speech, and Music Processing

METHODOLOGY Open Access

DeepDet: YAMNet with BottleNeck Attention Module (BAM) TTS synthesis detection

Rabbia Mahum[1*], Aun Irtaza[1], Ali Javed[2], Haitham A. Mahmoud[3] and Haseeb Hassan[4]

Abstract

Spoofed speeches are becoming a big threat to society due to advancements in artificial intelligence techniques. Therefore, there must be an automated spoofing detector that can be integrated into automatic speaker verification (ASV) systems. In this study, we recommend a novel and robust model, named *DeepDet*, based on deep-layered architecture, to categorize speech into two classes: spoofed and bonafide. *DeepDet* is an improved model based on Yet Another Mobile Network (YAMNet) employing a customized MobileNet combined with a bottleneck attention module (BAM). First, we convert audio into mel-spectrograms that consist of time–frequency representations on mel-scale. Second, we trained our deep layered model using the extracted mel-spectrograms on a Logical Access (LA) set, including synthesized speeches and voice conversions of the ASVspoof-2019 dataset. In the end, we classified the audios, utilizing our trained binary classifier. More precisely, we utilized the power of layered architecture and guided attention that can discern the spoofed speech from bonafide samples. Our proposed improved model employs depth-wise linearly separate convolutions, which makes our model lighter weight than existing techniques. Furthermore, we implemented extensive experiments to assess the performance of the suggested model using the ASVspoof 2019 corpus. We attained an equal error rate (EER) of 0.042% on Logical Access (LA), whereas 0.43% on Physical Access (PA) attacks. Therefore, the performance of the proposed model is significant on the ASVspoof 2019 dataset and indicates the effectiveness of the *DeepDet* over existing spoofing detectors. Additionally, our proposed model is robust enough that can identify the unseen spoofed audios and classifies the several attacks accurately.

Keywords Deep learning, Spoofing detector, Fake speech detection

1 Introduction

Speech is commonly used as a transmitting medium in digital devices such as mobile phones and computers. Some other characteristics of speech exist, i.e., rhythm, genre, pitch, etc. However, with the advancement of artificial intelligence and deep learning models [1, 2], it has become easy to manipulate the signals and generate fake speech to deceive the listener. Moreover, various speech synthesis algorithms, i.e., GAN [3], Deepvoice [4], cotton [5], and wavelet [6], have gained importance to generate natural speech just like humans and defeat the automatic speaker verification (ASV) systems. For example, false information related to politics based on deep fakes became a major threat to the US presidential elections in 2020 [7]. Furthermore, an incident of loss of USD 243,000 occurred when an audio-deep fake [8] was employed in bank transactions. Therefore, these incidents show the vulnerability of the ASV systems that are used widely in various security systems.

In the ASVspoof 2019 competition, the dataset has two partitions: logical access (LA) and physical access (PA) attacks. There exist several methods to synthesize speeches, including three main types: replay attack (RA),

*Correspondence:
Rabbia Mahum
rabbia.mahum@uettaxila.edu.pk
[1] Computer Science Department, UET Taxila, Taxila, Pakistan
[2] Software Engineering Department, UET Taxila, Taxila, Pakistan
[3] Industrial Engineering Department, College of Engineering, King Saud University, 11421 Riyadh, Saudi Arabia
[4] College of Big Data and Internet, Shenzhen Technology University (SZTU), Shenzhen, China

text-to-speech synthesis (TTS), and voice cloning (VC), which are considered in the logical access (LA) set of the ASVSpoof2019 dataset, and physical access (PA) attacks include recordings that simulate attacks on physical access systems. These attacks may involve impersonation, replay, or other methods aimed at deceiving the verification system respectively. Researchers have proposed different approaches [9–11] for spoofing detection. Some algorithms exist based on machine learning techniques to discern the audio based on data-driven and knowledge-focused countermeasures [12–14]. However, in traditional machine learning algorithms, hand-crafted feature extraction is performed, which is a consuming task and more complex due to the need for feature engineering to select optimal features. Whereas in the deep learning model, features are extracted and selected automatically and are more flexible.

With the advancement in the domain of Convolutional Neural Networks(CNNs), some methods have been proposed based on deep layers, such as Chintha et al., who proposed a recurrent CNN structure to detect spoofed (fake) audio [15]. Moreover, a lightweight convolutional neural network, namely LCNN, has been employed by [16], utilizing the softmax loss function to detect anti-spoofed attacks. Furthermore, various combinations of detecting systems have been tested along with ResNet [17] and explored with other classifiers as well for better performance [18, 19]. In [20], a model was employed based on an end-to-end ensemble method to learn the fusions of various detection systems. Even though the performance of these proposed algorithms was satisfactory, there exists an issue of generalization for unseen attacks on the models. Therefore, it is necessary to introduce an efficient and robust system that can carry out the detection of fake audio from any source.

The model proposed in this study focuses on the detection of synthesized speech rather than real due to two major concerns: (1) defining a comprehensive model for detecting all possible variations of genuine audio can be extremely complex and may lead to false positives or false negatives, and (2) adversaries can employ various techniques to generate fake audio that mimics genuine recordings, making it challenging to rely solely on detecting genuine audio. Therefore, focusing on fake audio detection allows system designers to address potential threats and vulnerabilities introduced by malicious actors attempting to deceive the system.

This study proposes a novel and robust framework to detect spoofed voices, such as LA and PA attacks, based on a deep learning model, namely *DeepDet.* Our proposed model is mainly divided into three phases. First, audio features have been extracted in the form of images known as mel-spectrograms. Second, a deep layered network has been trained using the ASVspoof-2019 dataset to classify the audio input as fake or real. Third, the network performs the binary classification of mel-spectrograms. To evaluate our proposed system's performance and effectiveness, we perform our experiments utilizing the publically available dataset, i.e., ASVspoof 2019. More precisely, we proposed a robust system that identifies and classifies the spoofed audios generated by the text-to-speech and cloning algorithms-based systems. We assessed the performance of the suggested system using PA (replay and bonafide samples) and LA (voice conversion, speech synthesis, and bonafide) sets from the ASVspoof 2019 corpus. The major offerings of the proposed model are presented below:

- To propose a novel deep learning based model, namely *DeepDet* based on improved YAMNet's architecture for spoofed audio detection similar to image classification models.
- The *DeepDet* employs an improved architecture using an attention module for the feature extraction from mel-spectrograms. It employs depth-wise separable convolutions, that is why our proposed model is lightweight.
- An attention block makes our model focus on relevant parts of the input, reducing the information loss and improving the model's ability to capture fine-grained patterns.
- Our proposed method is a robust speech spoofing detector that can be utilized to detect unseen synthetic voice attacks along with replay attacks and voice conversion.
- We evaluated our proposed system by employing extensive experiments that confirm the significance of our proposed system over existing techniques.

The remaining paper is ordered as follows: Sect. 2 defines the related work, Sect. 3 enlightens the methodology of the proposed technique, Sect. 4 defines the experiments performed, and Sect. 5 demonstrates the conclusion and limitations.

1.1 Related work

With the advancement of technology and electronic devices, the processing of content based on speech, such as music, ambient sounds, games, and entertainment, has become a significant field for researchers. Various models have been proposed for the classification of audio based on audio features [21–25]. Moreover, in the last two decades, text-to-speech systems have become so powerful that they are capable of generating a realistic voice after training limited audio samples from target speakers [26]. Therefore, it is a huge threat for ASV systems as they may attacked by the naturalness of the speech generated [27]. The applications that can protect the ASV systems from attacks are called deepfake speech detectors. Thus, various machine learning and deep learning-based works have been proposed for the detection of forged speech.

In [28], a support vector machine (SVM)-based classifier has been utilized as AVS employing GMM. They attained an equal error rate of 4.92% and 7.78% on the 2006 NIST for speaker identification core test. The authors have proposed the Gaussian Mixture Model (GMM) and a relative phase shift with a support vector machine (SVM) for synthetic speech detection to minimize the weaknesses of speaker verification systems. Moreover, a detailed comparison of the Hidden Markov Model (HMM) and DNN has been performed for the detection of spoofed speech [29]. In [30], the proposed model employs the spectrograms in image form as input to CNN, thus forming a base of audio processing using images. In [31], various feature descriptors have been used, such as Mel Frequency Cepstral Coefficient (MFCC), spectrogram, etc., and the effect of GMM-UBM on the accuracy has been analyzed. It is concluded that the combination of different feature descriptors gives better results in terms of equal error rate (EER). Chao et al. [28] utilized SVM to discern the real speeches from the fake recordings of the claimed man. Similarly, in [32], Chao has employed two core methods, Kernel Fisher Discriminant (KFD) and SVM, to verify speakers and attained better results as compared to their previous work based on the GBM and UBM methods. Moreover, to decline the computational cost of the polynomial kernel SVM by exchanging the dot product among two utterances with two i-vectors [33]. Furthermore, the authors applied the features selection technique, attaining a 64% dimensionality reduction in features with an equal error rate of 1.7% [33]. Whereas Loughran et al. [34] overcame the issue of imbalanced data (where one class sample is greater than the other) by utilizing a genetic algorithm (GA) with an adjusted cost function. Malik et al. [35] developed a system for audio forgery detection based on acoustic signatures of the environment by investigating the integrity of audio. However, these proposed models failed to address synthesized audio content with high precision.

In [36], a DNN-based classifier has been proposed to detect and employ highlight Human Log Likelihoods (HLL) as a metric for scoring and proved to be better than classical log-likelihood ratios (LLR). Additionally, they also utilized various cepstral coefficients for the classifier's training [37, 38] also employed a convolutional neural network for the audio classification. An extensive comparison has been made using DL techniques for fake audio detection in [39], demonstrating that CNN and recurrent neural network (RNN) based models give better results than all other employed techniques [40] explains that the spectral features are significant to use for the detection of synthetic speech. For example, MFCC features are better than other spectral features for the model's input. Furthermore, [41] describes the challenges and limitations of the spoofed detection models. In [42], a bispectral method for the analysis and detection of synthetic voices has been proposed. They examined uncommon spectral features in fake speeches synthesized using DNNs, which they called bispectral features. They also tried to find high-order polyspectral features to discern the fake audio. A capsule network-based approach has been proposed in [43]. They enhanced the generalization of the proposed system and examined the artifacts deeply to increase the overall performance of the model. They also investigated the replay attacks in audios employing their network. In [44], authors proposed a model for fake audio detection named *DeepSonar.* They analyzed the network layers and the activation patterns for various input audios to examine the difference between fake and real speeches. They employed three datasets consisting of English and Chinese language and attained an average accuracy of 98.1%. In [45], authors have proposed a model for fake audio detection based on micro-features such as voicing onset timing (VOT) and coarticulation. They analyzed that VOT numbers are high in fake speeches and attained a 23.5% error rate employing a fusion of both feature descriptors. The authors claimed that these micro-features can be used as standalone features for fake audio detection. Moreover, temporal convolutional networks (TCN) [46] have outperformed traditional algorithms such as RNNs and LSTMs for various tasks.

The latest deep learning techniques for text-to-speech synthesis systems, such as [47], clone the voice using original speech recordings. It requires a few minutes of recording in real voice and generates fake audio in some seconds. Although the techniques have been improved [48], they still face the challenge of naturalness. The authors in [49] have investigated the usage of RAWNet2 for the spoofing detection. They improved the architecture of RawNet2 and proved that their results are second best for the detection of A17 attacks. The authors proposed a novel feature extraction process in [50] for replay attack detection. The developed l Cochlear Filter Cepstral Coefficients-based Instantaneous Frequency using Quadrature Energy Separation Algorithm (CFCCIF-QESA) features, with excellent temporal resolution as well as relative phase information [51] suggested integrating orthogonal convolution into RawNet for fake audio detection, which serves to decrease the correlation between filters when optimizing Sinc-conv parameters, thereby enhancing discriminability. Additionally, they introduced temporal convolutional networks (TCN) to capture long-term dependencies in speech signals. Experimental results on the ASVspoof 2019 dataset reveal that their model, namely TO-RawNet, demonstrates a relative reduction of 66.09% in equal error rate (EER) on the logical access scenario compared to RawNet. This underscores the effectiveness of the approach in detecting fake audio attacks.

The existing studies have failed to fully discern the fake voices, and thorough evaluation has not been performed to evaluate their robustness employing the various manipulated voices (changing pitch, rhythm, and resampling it without changing the linguistics). Furthermore, audio artifacts are more difficult to detect than image artifacts as transforming audio signals into a frequency representation (mel-spectrograms) facilitates pattern identification by models. The temporal and spectral information is also considered to capture the audio's frequency content. In addition to this, as indoor or outdoor voices have environmental noises, it is very easy for fake voice generators to add real-world noise to the voice to fool the listener or the ASV system. Thus, an automatic fake audio detector that is robust enough to identify the fake audio of various environments is still needed.

2 Methodology

Deep learning architectures are made of various layers, such as input, hidden, and classification layers, as shown in Fig. 1. These hidden layers have various types, i.e. convolutional, batch normalization, pooling, activation, etc. The deep learning models extract features utilizing various filters convolving over the input images. Moreover, when the filters are convolved over all the data, then a feature map is formed. These feature maps are reduced in dimensions employing pooling layers, minimizing the computational power of the system. These feature maps can be fed again to further convolution layers, repeating the above steps.

Numerous applications exist for various purposes, such as facial feature recognition [52], speech identification[45], and emotion [53]. The presented system consists of three main phases, i.e., (1) features extraction, (2) training, and (3) classification. We employed features extraction utilizing a feature extraction layer through which mel-spectrograms have been generated and passed to an improved architecture of MobileNet [54] guided by the attention module as a base network in YamNet [55]. The audios comprised a 16000 Hz sampling frequency

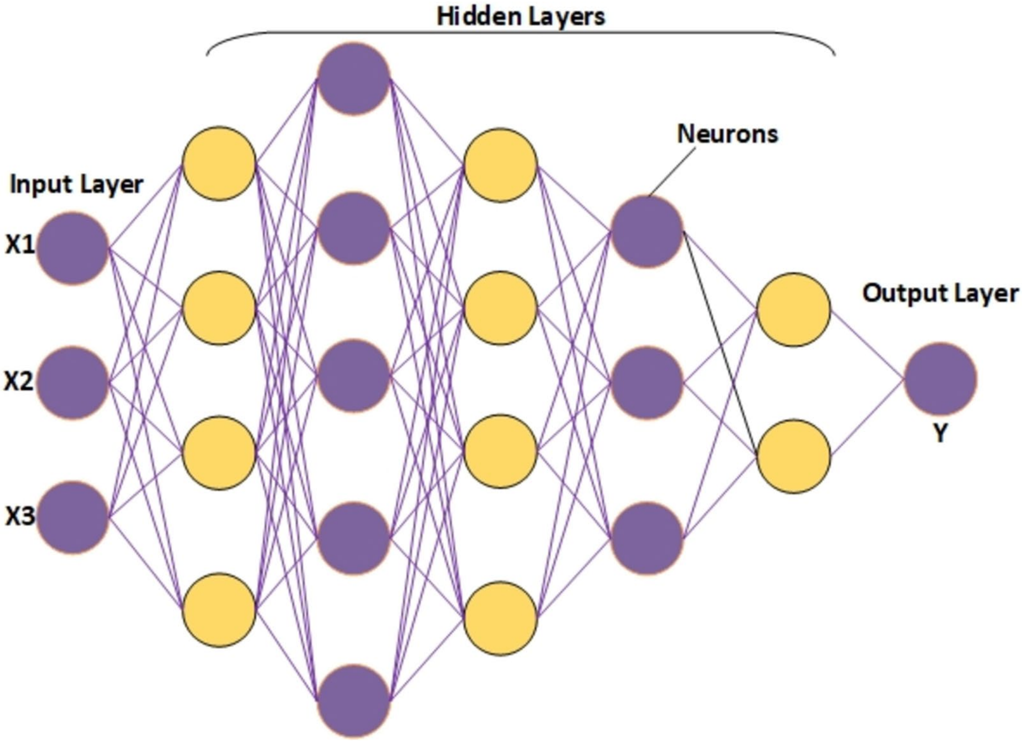


Fig. 1 General architecture of deep learning model

to capture the essential characteristics of speech. It provides a balance between capturing the necessary frequency content for clear speech communication and minimizing bandwidth requirements. Then, the audio was transformed into mel-spectrograms of 96×64 size. Secondly, we trained an improved network, i.e., *Deep-Det* over the generated mel-spectrograms belonging to two classes such: as Bonafide (Real) and Spoofed (Fake). The Mel-spectrograms of fake audio are different from real audio. Therefore, the proposed system learns the patterns precisely for two classes. Thirdly, we classified various input audios using the trained classifier. Moreover, the MobileNet employs linearly separable depth-wise convolutions. Therefore, our proposed model becomes lightweight. The design of the proposed system is shown in Fig. 2.

In the improved version, we customized the base network MobileNetV1, adding three layers grouped convolutional 2D layer, instance batch normalization layer, and activation layer before the fully connected layer. Moreover, we have changed all the batch normalization layers in the original network to instance normalization, which improves the convergence of the network. We increased the depth of the proposed network while decreasing the sensitivity of the model to the hyperparameters. Our proposed model extracts the most representative features from the mel-spectrograms generated from audio. Additionally, our model is lightweight due to depth-wise separable convolutions.

2.1 YAMNet architecture

Transfer learning is a famous deep learning aspect in which the model can learn the features from any other trained model. The key aspect of transfer learning is to minimize the computational cost of utilizing previously learned patterns. It is preferred to employ the transfer learning concept when a large size of unlabeled data is available to train a model. Therefore the pre-trained model utilizes its previous training features to reduce the time and effort. YAMNet employs the MobileNetV1 as the base network and is a pre-trained model on the Google AudioSet dataset for 521 audio events. Therefore, in our work, we are training YAMNet on unbalanced data and achieving significant performance. Before the features extraction phase, resampling is performed into 16,000 Hz with one channel audio. Moreover, YAMNet is a DL-based model. Therefore it extracts the audio features automatically due to the feature extraction layer. The feature extraction layer extracts the audio features in the form of spectrograms, and then these spectrograms are fed to improved MobileNet layers for classification. The layered architecture of the original YAMNet is shown in Fig. 3.

2.2 MobileNet

The MobileNet is developed using depth-wise separable convolutions that factorize a simple convolution into depth-wise convolution, as well as a 1×1 convolution that is identified as a point-wise convolution. These depth-wise convolutions employ one filter to each channel, splitting the standard convolution into two separable layers, one to apply filter and the other for concatenation. Furthermore, the point-wise convolution is employed of 1×1 size and concatenates the output with the depth-wise convolution. Due to this factorization, the size and computational complexity are decreased significantly. The standard convolution, along with depth-wise and point-wise convolution, are shown in Fig. 4. An input as D_k x D_k x M feature map k is fed to the standard convolutional layer, and the output feature map is generated as G. Here, D_k represents the spatial height and width of an input feature map, M represents the total number of channels as input, D_G refers to the spatial height and width of the feature map in output, and N represents the total output channels.

The standard convolutional layer is represented by conv. Kernel as L having a size of $D_L \times D_L \times M \times N$. Here, D_L refers to the dimension in the spatial context of a square kernel, M represents the total input channels, and N refers to the total output channels. Suppose we employ stride 1, and padding, therefore the output feature map is calculated as below:

$$G_{L,l,n} = \sum_{i,j,m} .k_{L+i-1,L+j-1,m} \tag{1}$$

Moreover, the standard convolutional operation involves the cost as below:

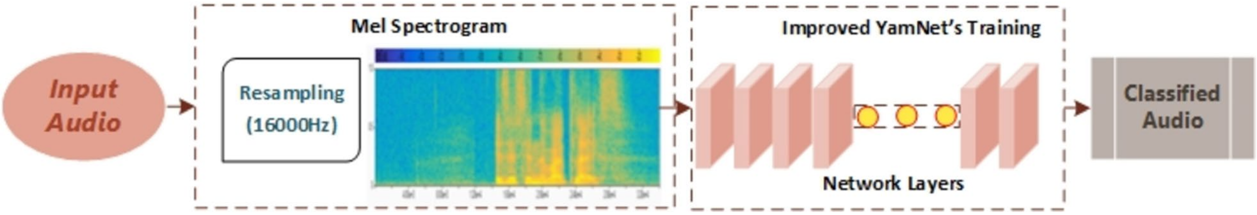


Fig. 2 Flow diagram of the proposed system

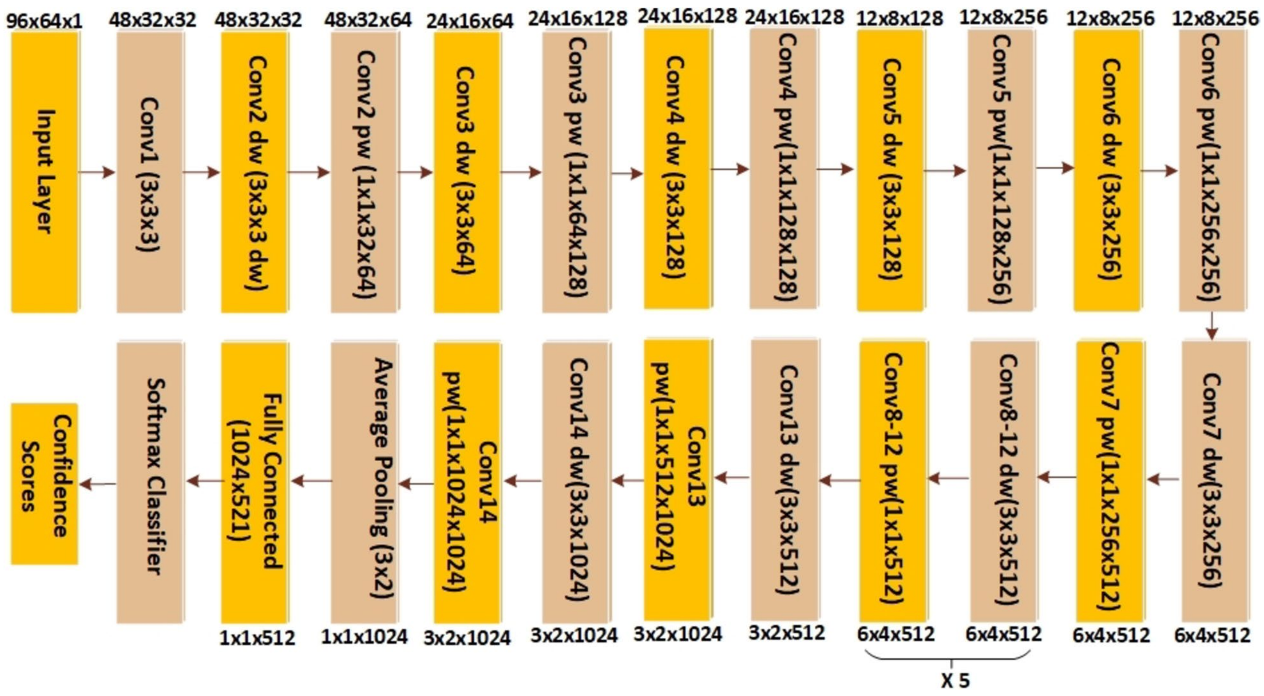


Fig. 3 Architecture of original YAMNet

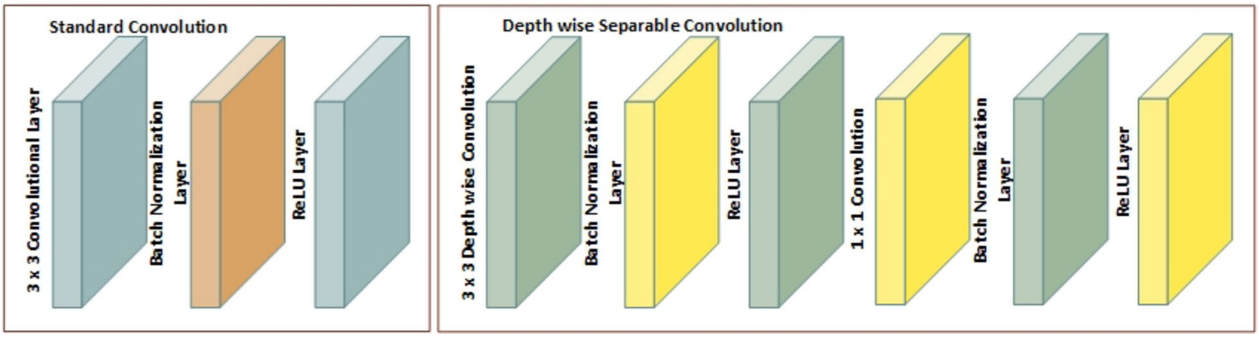

Standard Convolution
3 x 3 Convolutional Layer
Batch Normalization Layer
ReLU Layer
Depth wise Separable Convolution
3 x 3 Depth wise Convolution
Batch Normalization Layer
ReLU Layer
1 x 1 Convolution
Batch Normalization Layer
ReLU Layer

Fig. 4 Standard 3×3 convolution Vs depth wise convolution

$$D_L.D_L.M.N.D_k.D_k \tag{2}$$

Here, the cost of computation relies upon the product of the total input channels as *M*, the total output channels as *N*, and the kernel size as D_L and the size of the feature map as $D_k.D_k$. Furthermore, the Mobile network considers all these computational terms and their respective connections. It breaks the connections employing depth-wise separable convolutions among the output channel and the kernel size. The standard convolutional functions utilize the filtering and combining operations on features using the convolutional kernels, providing the new output of feature representations. Moreover, these two steps, such as filtering and combining, can be divided into two separate processes via depth-wise separable convolutions to minimize the computational cost.

Depth-wise separable convolution consists of 2 layers, i.e., pointwise convolution and depth-wise convolution. We utilize depth-wise convolution to employ a unit filter to all input channels. Whereas, point-wise convolution is applied using 1×1 convolution to make a linear combination of the output from the depth-wise layer. Moreover, mobile networks use two more layers, batch normalization, and ReLU non-linearity, for both types of layers. Batch normalization layers are usually employed in convolutional neural networks. The output of batch normalization (BN) is made of 4-D tensors, which are referred to as $I_{b,c,x,y,}$ and $F_{b,c,x,y}$ correspondingly. Where *b* represents batch, *c* is the channel, and *x and y* are two spatial dimensions, respectively. When an input is a form of images, then channels are based on RGB channels. BN layers employ a similar normalization in each channel for all activations.

$$I_{b,c,x,y} = \gamma_c \frac{I_{b,c,x,y} - \mu_c}{\sqrt{\sigma_c^2 + \in}} + \beta_c \forall \leftrightarrow b, c, x, y \tag{3}$$

$$\mu_c = \frac{1}{|\varnothing|} \sum_{b,x,y} I_{b,c,x,y} \tag{4}$$

In Eq. 3, BN deducts the mean activation $\mu_{c,}$ as shown in the below equation. It is subtracted from all input activations of channel *c* and $\varnothing$ comprises channel c activations across all mini-batches features *b* and spatial locations *x,y.* Moreover, the centered activation is divided by the standard deviation σ_c and added to $\in$ for the stability in computation. The normalization in BN is followed by various affine transformations using and β_c in each channel during training. The significance of employing BN is that it evades activation explosion by improving all the activations to make them zero-mean. Due to this, it becomes possible to train a network employing large learning rates, as the means and variances have been normalized. Therefore activations should not grow uncontrollably. Furthermore, large learning rates allow the algorithm to reach the convergence point fast. Small learning rates show slight progress in flat directions of the optimization and may converge at the sharp local minimum, exhibiting less generalization performance [56].

Rectified linear units (ReLU), referred to as an activation function, first found importance in acoustic models [57] exhibiting mathematical and biological characteristics. It is described as a source of improved training processes of deep learning models. It works based on threshold values at 0, such as *f(x)=max(0,x).* It gives output as 0 when *x* is less than 0 and provides a linear function when *x* is larger than or equivalent to 0 i.e., $x \geq 0$. The improved YAMNet, i.e., *DeepDet,* is exhibited in Fig. 5.

The rectified linear unit (ReLU), which is an activation function, yields 0 as an output where $x < 0$ and then yields a linear having a slope of 1 where $x > 0$.

We employ the ReLU activation function among all hidden layers of a deep neural network and as the classification function in the output layer of the proposed network.

The depth-wise convolution having one filter for a single input channel is computed as shown in the equation below.

$$\}_{L,l,m} = \sum_{i,j} \mathcal{L}_{i,j,m}.K_{L+i-1,L+j-1,m} \tag{5}$$

Here, $\mathcal{L}$ represents the depth-wise convolutional kernel having size D_L x D_L x *M; here,* the m^{th} filter in $\mathcal{L}$ is employed on the m^{th} channel in *K* to form an output feature vector as $\}$. The cost for computation of depth-wise convolution is computed as below:

$$D_L.D_L.M.D_K, \tag{6}$$

Moreover, the depth-wise convolution is more proficient than the standard convolution. However, it employs filtering only on input channels and does not merge to form new features. Therefore, an additional layer is required that can combine the depth-wise convolution's output using 1×1 convolution to form new features. The combined form of depth-wise and point-wise convolution is known as depth-wise separable convolution, which was first introduced by [58]. The summation of depth-wise and point-wise 1×1 convolutions can be represented mathematically as below:

$$M.D_L.D_L.D_K.D_K + M.N.D_k \tag{7}$$

We can express convolution in two steps, i.e., filtering and combining mathematically in Eq. 8.

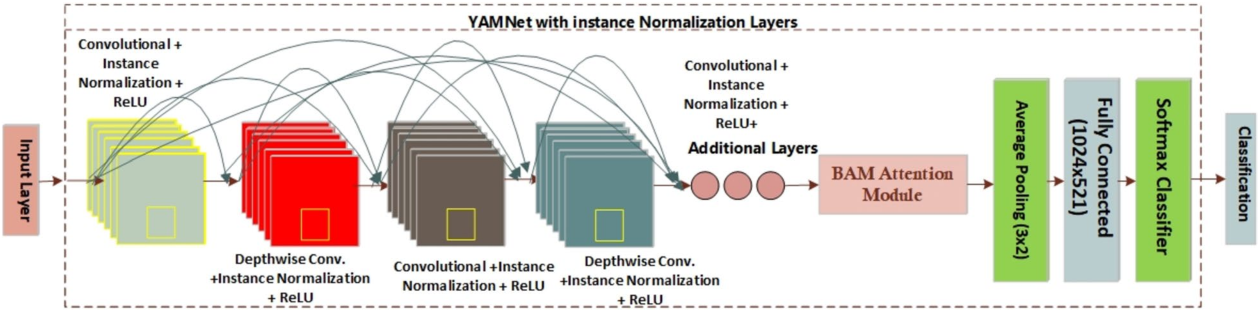

YAMNet with instance Normalization Layers
Input Layer
Convolutional + Instance Normalization + ReLU
Depthwise Conv. +Instance Normalization + ReLU
Convolutional +Instance Normalization + ReLU
Depthwise Conv. +Instance Normalization + ReLU
Convolutional + Instance Normalization + ReLU+
Additional Layers
BAM Attention Module
Average Pooling (3x2)
Fully Connected (1024x521)
Softmax Classifier
Classification

Fig. 5 Improved YAMNet's architecture

$$\frac{M.D_L.D_L.D_K.D_K + D_k.D_k.M.N}{D_k.D_k.D_L.D_L.M.N} = \frac{1}{N} + \frac{1}{D_L^2} \tag{8}$$

The mobile network employs 3×3 convolutions that require approximately eight times less computational cost than standard convolutions.

2.3 Improved architecture

The MobileNet is developed using depth-wise separable convolutions as described in the previous unit, excluding the first fully convolutional layer. The architecture of our improved MobileNet is shown in Table 1. It is depicted that we have added three extra layers before the global average pooling layers, such as the grouped convolutional 2D layer, instance batch normalization layer, and ReLU layer. We increased another block of three layers to increase the efficiency of the model by extracting the most representative features from the mel-spectrograms. Moreover, we have changed all the batch normalization layers with instance normalization layers [59]. They operate differently on input data. While instance normalization (IN) transforms an individual training sample, Batch Normalization (BN) applies the transformation to the entire mini-batch of samples. This makes BN reliant on the batch size, as a larger batch size is necessary to obtain a statistically more accurate mean and variance. Implementing a large batch size can be challenging due to memory constraints. Consequently, when faced with memory limitations, smaller batch sizes may be chosen, which can pose problems in certain situations. The use of a very small batch size can introduce training errors because the mean and variance become more prone to noise. IN outperforms BN in scenarios where a small batch size is employed.

Moreover, because BN's effectiveness is tied to the batch size, it cannot be applied in the same manner during test time as it is during training. This limitation arises because, typically, only one example is processed during the testing phase, making it impossible to compute mean and variance in the same way as during training. Instead, BN utilizes moving averages and variance for inference during test time. In contrast, IN is independent of the batch size, ensuring consistent implementation for both training and testing. Besides this, batch normalization introduces additional noise during training since the outcome for a specific instance is influenced by neighboring instances. Interestingly, this type of noise can have both positive and negative effects on the network.

More precisely, a mel-spectrogram in the form of a 2D image having dimensions of 96×64×1 is fed from the image input layer to the first convolutional 2D layer. Then, the convolutional 2D layer gives an output of 48×32×32 channels having stride two and the same padding. In 3rd step, instance normalization is employed with 32 channels having offset 1×1×32 and scale 1×1×32. In the 4th step, the ReLU activation function is employed, which gives 48×32×32 activations. At the 5th step, depth-wise 2d convolution is employed on 32 groups of 1 3×3×1 convolutions having learnable as weights: 3×3×1×1×32, bias: 1×1×1×32 with stride one and padding same. At the 6th and 7th steps, instance normalization and activation function ReLU is employed, giving activations as 48×32×32. Similarly, this sequence is followed till the last activation function at step 85, giving output 3×2×1024. As described before, when audio files are converted into mel-spectrograms, the number of bands is 64.

2.4 BAM attention module

Channel dependency usage is a significant way of improving CNN model execution. To increase the performance of state-of-the-art models with negligible computational cost, we used attention block, i.e., BottleNeck Attention Module (BAM) [60], in our improved YAMNet model, as shown in Fig. 5. BAM's architecture depends on two pathways: spatial and channel. It gets training in an end-to-end way with our proposed DeepDet. Different channel weights are trained using the cost function, and the weight coefficients of the feature channel are obtained automatically. The attention module assists the model in attaining intermediate features more effectively. The proposed attention module's architecture is shown in Fig. 6. The F represents the feature map, whereas M(F) is an attention map computed by the module using two attention methods: spatial as M_s and channel as M_c. Two hyperparameters exist, i.e., r as the reduction ratio and d as the dilation value. More specifically, r controls the overhead among both attention methods, and d spatially helps in contextual information using the receptive field size.

The incorporation of the attention mechanism in BAM aids in acquiring more informative and distinctive representations. This, in turn, improves the extraction of features and fosters a deeper comprehension of intricate patterns in the data, ultimately augmenting the network's capacity for learning and generalization. Additionally, BAM plays a role in enhancing the robustness of a neural network by empowering it to dynamically adjust its attention in response to the input context. This adaptability equips the network to effectively manage variations and shifts in the input data, bolstering its resilience to diverse conditions and scenarios. Attention mechanisms, exemplified by BAM, have the potential to mitigate computational overhead by allowing the network to concentrate on pertinent segments of the input, thereby conserving computational resources.

Table 1 Layer-wise details of our proposed MobileNet

Type	Activations	Learnable	Stride/channel	Total learnable
Image input	96×64×1	–	–	0
Convolution 2D (Conv)	48×32×32	Weights: 3×3×1×32 Bias: 1×1×32	32 3×3×1 convolutions Stride: [2 2] Padding: same	320
Instance normalization	48×32×32	Offset: 1×1×32 Scale: 1×1×32	32 Channels	64
ReLU	48×32×32	–	–	0
Grouped convolution depthwise (GConv DW)	48×32×32	Weights: 3×3×1×1×32 Bias: 1×1×1×32	32 groups of 1 3×3×1 convolutions Stride: [1 1] Padding: same	320
Instance normalization	48×32×32	Offset: 1×1×32 Scale: 1×1×32	–	64
ReLU	48×32×32	–	–	0
Conv	48×32×64	Weights: 1×1×32×64 Bias: 1×1×64	64 1×1×32 convolutions Stride: [1 1] Padding: same	2112 128 0
GConv DW	24×16×64	Weights: 3×3×1×1×64 Bias: 1×1×1×64	64 groups of 1 33×1 Convolutions Stride: [2 2] Padding: same	640 128 0
Conv	24×16×128	Weights: 1×1×64×128 Bias: 1×1×128	128 1×1×64 Convolutions Stride: [1 1] Padding: same	8320 256 0
GConv DW	24×16×128	Weights: 3×3×1×1×128 Bias: 1×1×1×128	128 groups of 1 3×3×1 Convolutions Stride: [1 1] Padding: same	1280 256 0
Conv	24×16×128	Weights: 1×1×128×128 Bias: 1×1×128	128 1×1×128 Convolutions Stride: [1 1] Padding: same	16,512 256 0
GConv DW	12×8×128	Weights: 3×3×1×1×128 Bias: 1×1×1×128	128 groups of 1 3×3×1 Convolutions Stride: [2 2] Padding: same	1280 256 0
Conv	12×8×256	Weights: 1×1×128×256 Bias: 1×1×256	256 1×1×128 Convolutions Stride: [1 1] Padding: same	33,024 512 0
GConv DW	12×8×256	Weights: 3×3×1×1×256 Bias: 1×1×1×256	256 groups of 1 3×3×1 convolutions Stride: [1 1] Padding: same	2560 512 0
Conv	12×8×256	Weights: 1×1×256×256 Bias: 1×1×256	256 1×1×256 Convolutions Stride: [1 1] Padding: same	65,972 512 0
GConv DW	6×4×256	Weights: 3×3×1×1×256 Bias: 1×1×1×256	256 groups of 1 3×3×1 Convolutions Stride: [2 2] Padding: same	2560 512 0
Conv	6×4×512	Weights: 1×1×256×512 Bias: 1×1×512	512 1×1×256 Convolutions Stride: [1 1] Padding: same	131,584 1024 0
GConv DW	6×4×512	Weights: 3×3×1×1×512 Bias: 1×1×1×512	512 Groups Of 1 3×3×1 Convolutions Stride: [1 1] Padding: same	5120 1024 0
Conv	6×4×512	Weights: 1×1×512×512 Bias: 1×1×512	512 1×1×512 convolutions Stride: [1 1] Padding: same	262,656 1024 0
GConv DW	6×4×512	Weights: 3×3×1×1×512 Bias: 1×1×1×512	512 groups of 1 3×3×1 convolutions Stride: [1 1] Padding: same	5120 1024 0
Conv	6×4×512	Weights: 1×1×512×512 Bias: 1×1×512	512 1×1×512 convolutions Stride: [1 1] Padding: same	262,656 1024 0

Table 1 (continued)

Type	Activations	Learnable	Stride/channel	Total learnable
GConv DW	6×4×512	Weights: 3×3×1×1×512 Bias: 1×1×1×512	512 groups of 1 3×3×1 convolutions Stride: [1 1] Padding: same	5120 1024 0
Conv	6×4×512	Weights: 1×1×512×512 Bias: 1×1×512	512 1×1×512 convolutions Stride: [1 1] Padding: same	262,656 1024 0
GConv DW	6×4×512	Weights: 3×3×1×1×512 Bias: 1×1×1×512	512 groups of 1 3×3×1 convolutions Stride: [1 1] Padding: same	5120 1024 0
Conv	6×4×512	Weights: 1×1×512×512 Bias: 1×1×512	512 1×1×512 convolutions Stride: [1 1] Padding: same	262,656 1024 0
GConv DW	6×4×512	Weights: 3×3×1×1×512 Bias: 1×1×1×512	512 groups of 1 3×3×1 convolutions Stride: [1 1] Padding: same	5120 1024 0
Conv	6×4×512	Weights: 1×1×512×512 Bias: 1×1×512	512 1×1×512 convolutions Stride: [1 1] Padding: same	262,656 1024 0
GConv DW	3×2×512	Weights: 3×3×1×1×512 Bias: 1×1×1×512	512 groups of 1 3×3×1 convolutions Stride: [2 2] Padding: same	5120 1024 0
Conv	3×2×1024	Weights: 1×1×512× ×1024 Bias: 1×1×1024	1024 1×1×512 convolutions Stride: [1 1] Padding: same	525,312 2048 0
GConv DW	3×2×1024	Weights: 3×3×1×1×1024 Bias: 1×1×1×1024	1024 groups of 1 3×3×1 convolutions Stride: [1 1] Padding: same	10,240 2048 0
Conv	3×2×1024	Weights: 1×1×1024×1024 Bias: 1×1×1024	1024 1×1×1024 convolutions Stride: [1 1] Padding: same	1,049,600 2048 0
Conv	3×2×1024	Weights: 1×1×1024×1024 Bias: 1×1×1024	1024 1×1×1024 convolutions Stride: [1 1] Padding: same	1,049,600 2048 0
Avg. Pooling	1×1×1024	–	–	0
FC Layer	1×1×2	Weights: 2×1024 Bias: 2×1	–	2040
Softmax	1×1×2	–	Binary classifier	

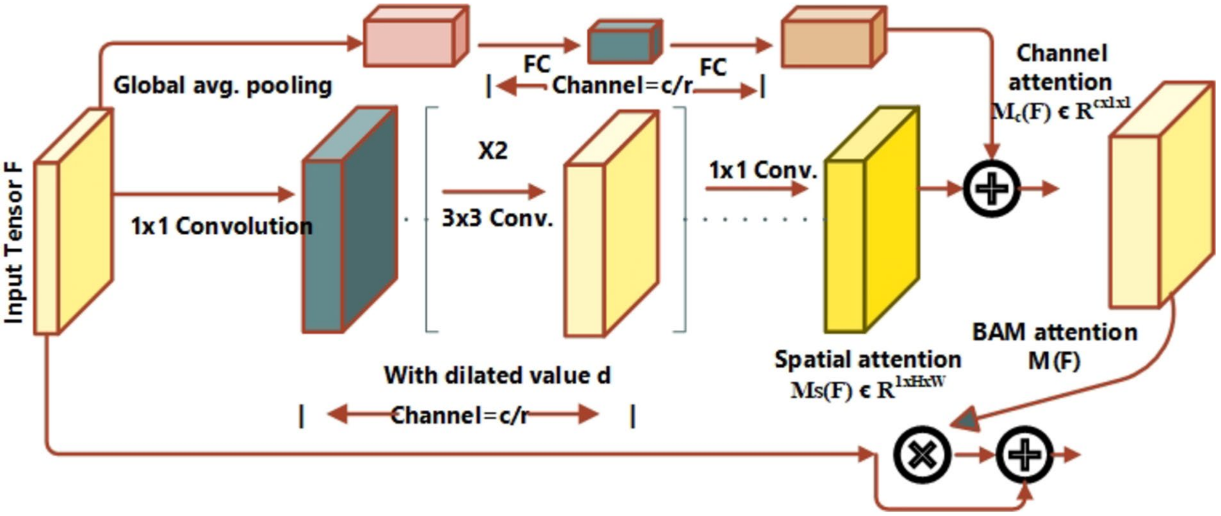
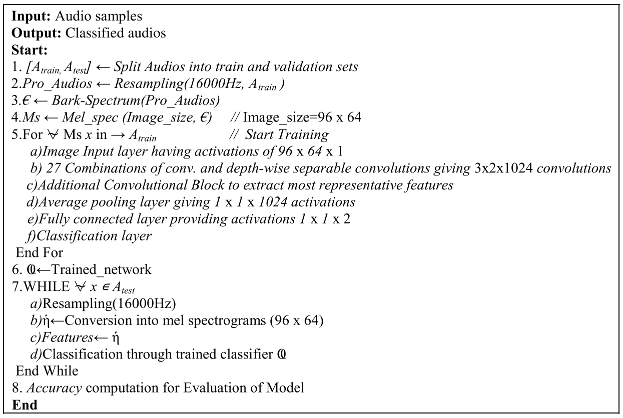


Fig. 6 BAM attention module

Algorithm 1. Steps for *DeepDet*



```
Input: Audio samples
Output: Classified audios
Start:
1. $[A_{train}, A_{test}]$ ← Split Audios into train and validation sets
2.Pro_Audios ← Resampling(16000Hz, $A_{train}$ )
3.€ ← Bark-Spectrum(Pro_Audios)
4.Ms ← Mel_spec (Image_size, €)     // Image_size=96 x 64
5.For ∀ Ms $x$ in → $A_{train}$                    //  Start Training
    a)Image Input layer having activations of 96 x 64 x 1
    b) 27 Combinations of conv. and depth-wise separable convolutions giving 3x2x1024 convolutions
    c)Additional Convolutional Block to extract most representative features
    d)Average pooling layer giving 1 x 1 x 1024 activations
    e)Fully connected layer providing activations 1 x 1 x 2
    f)Classification layer
 End For
6. ⱷ←Trained_network
7.WHILE ∀ $x \in A_{test}$
    a)Resampling(16000Hz)
    b)ή←Conversion into mel spectrograms (96 x 64)
    c)Features← ή
    d)Classification through trained classifier ⱷ
 End While
8. Accuracy computation for Evaluation of Model
End
```

Table 2 Statistics of ASVSpoof 2019 LA and PA sets

Set	Logical access		Physical access	
	Spoofed	Bonafide	Spoofed	Bonafide
Train	22,800	2580	48,600	5400
Evaluate	63,882	7355	116,640	18,090
Dev	22,296	2548	24,300	5400
Total	36,326	12,483	189,540	28,890

Table 3 System specifications for the employed model

Hardware	Specifications
Computer	GPU Server
CPU	Intel Core i5
RAM	16 GB
GPU	NVIDIA GEFORCE GTX × 4

3 Experimental evaluation

3.1 Dataset

The challenge of spoofed voice detection came in 2015, known as the ASVSpoof 2015 corpus [61]. The aim was to develop a system to detect the synthesized or cloned speech and analyze the performance using the dataset samples. After 2 years, the ASVSpoof 2017 corpus [41] came into existence for the evaluation of the replay detection systems. A large and assorted dataset was introduced in 2019, known as ASVSpoof 2019 [62], comprising both logical access (LA) and physical access(PA) attacks. The first contained the voice conversion and synthesized speech samples, including bonafide audio. The later part consists of replay and bonafide audio samples. Real speech data is sourced from 107 speakers (46 male, 61 female) without notable channel or background noise influences. Spoofed speech is then created using various spoofing algorithms based on genuine data.

Furthermore, both parts have been split further into three sub-parts, namely, development, training, and evaluation sets. The logical access dataset comprises seventeen various text-to-speech and voice cloning systems. Moreover, these systems are trained using the voice cloning toolkit VCTK [63]. Among these systems, six have been labeled as known attacks, whereas the other 11 systems are known as anonymous attacks. The training and development audio samples are taken from known attacking systems, and evaluation samples are collected from 11 unknown and two known attacks. The Logical Access set consists of 2 VC systems that utilize spectral filter and artificial neural networks-based approaches. Furthermore, the LA set consists of 4 TTS systems that utilize artificial neural networks or concatenation of wave-form employing vocoders based on source-based filter Vocoder [64] or WaveNet Vocoder [65]. The 11 unknown spoofing systems consist of 2 VC, 6 TTS, and 3 Hybrid forms of VC and TTS systems utilizing various waveform-based methods such as GriffinLim [66], Neural waveform techniques [67], Generative adversarial networks (GAN) [68], and combinations of waveform and spectral filtering. The statistics of the ASVSpoof 2019 dataset are shown in Table 2, whereas a depth summary of the LA set is shown in Table 3.

Moreover, ASVspoof 2017 [41] comprises real replay speeches, while ASVspoof 2019 comprises synthesized replay recordings recorded under an acoustic environment to enrich the ASV system's reliability. Training and development recordings are produced, conferring to 9 replay and 27 acoustic configurations. The sizes of rooms are categorized as large, medium, and small rooms. All speeches are generated in various zones, such as A, B, and C, exhibiting varying distances (Da) between the talker and zone. The zone A voice quality is better than the B and C zones. Moreover, the eval recordings have been gathered in the same way as train and dev sets.

To evaluate the model on the LA set, we utilized the training samples as 25,000, including 2580 bonafide samples and 22800 spoofed samples to train *DeepDet.* We tested our model using both sets, that is, eval and dev sets. The eval set consists of 71,237 samples, including 7355 spoofed and 63882 bonafide samples, while the dev set consists of 24,844 samples, including 22,296 spoofed and 2548 bonafide samples. Furthermore, we have evaluated our proposed model using the PA dataset; we employed 54,000 samples, including 48,600 spoofed and 5400 bonafide samples, for the model's training. Additionally, we evaluated *DeepDet* using both remaining sets, that is, eval and dev sets of the PA database. The eval set comprises 134,730 samples, including 116,640 spoofed and 18,090 bonafide samples, and the dev set comprises 29,700 audios, including 24,300 spoofed and 5400 bonafide samples.

3.2 Environment

We performed the experiments using a GPU NVIDIA card, i.e., GEFORCE GTX with 4 GB memory. The details of the employed hardware are shown in Table 3. The operating system was Windows 10, which had 16 GB RAM. The experiment was performed on the Matlab 2021a.

3.3 Metrics

For the performance evaluation of the proposed model, we have utilized various metrics such as precision, recall, accuracy, equal error rate, and Tandem-detection cost function(t-DCF). Moreover, these metrics are relied on true positive (TP), false positive (FP), true negative (TN), and false-negative (FN). The TP refers to the correctly classified spoofed audios by our proposed model, FP refers to the number of audios that were incorrectly classified as spoofed, FN denotes the number of audios that were incorrectly classified as negative, i.e., bonafide, and TN refers to the number of audios that were correctly classified as a negative class such as bonafide. Furthermore, precision refers to the fraction of TP over the total audios (mel-spectrograms) classified as positive. The mathematical equation is given below.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{10}$$

The accuracy of the system indicates the correctly classified audio by the proposed system. The equation is presented below.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{11}$$

The recall is the fraction of the classified positive class audios to all spoofed audios whether they were classified as a real class by the system. The recall value closer to 1 refers to the better model. The equation of Recall is given below.

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{12}$$

Moreover, we employed an Equal Error Rate (EER) and t-DCF to analyze the performance of the proposed spoof detector. Suppose FAR (θ) and FRR (θ) refer to the false acceptance and rejection rates at threshold value θ, respectively. FRR and FAR decrease and increase monotonically at the rate of θ; therefore, the EER estimates the value of θ at which both FRR and FAR become equal.

$$\text{FRR}(\theta) = \frac{\text{count}\left(\text{human trials having score}\right) < \theta}{\text{Total human trials}} \tag{13}$$

$$\text{FAR}(\theta) = \frac{\text{count}\left(\text{spoofed trials having score}\right) > \theta}{\text{Total spoofed trials}} \tag{14}$$

The formula for t-DCF is given below.

$$t - \text{DCF} = \min\left\{\beta P_{miss}^{cm}(s) + P_{fa}^{cm}(s)\right\} \tag{15}$$

where the value of β is depends on application-specific parameters, such as priors and costs, as well as the performance of the ASV. $P_{miss}^{cm}(s)$ and $P_{fa}^{cm}(s)$ represent the miss and false alarm rates of the countermeasure system at the threshold s.

4 Performance over synthesized speech and voice conversion

In this experiment, we want to evaluate our proposed model, i.e., *DeepDet,* over text-to-speech synthesis (TTS) and voice conversions (VC) samples. Therefore, we employed an improved MobileNet as the base network of YAMNet to categorize the speeches into bonafide and spoofed speeches of TTS and voice conversion samples. The training dataset comprises genuine and falsified speech samples from 20 speakers (8 male, 12 female). In the datasets, there exist 4 TTS spoofed systems, including A01, A02, A03, and A04, whereas 2 VC spoofed methods, including A05 and A06, are utilized to generate spoofed samples for the LA dataset for training. The voice conversion systems include (1) neural network-based and (2) transfer function-based approaches. The speech synthesis systems were implemented using (1) waveform concatenation, (2) neural network-based parametric speech synthesis employing source-filter vocoders, and (3) neural network-based parametric speech synthesis utilizing Wavenet. In the developmental set, fake speech is produced using one of the spoofing algorithms employed in creating the training dataset. Meanwhile, in the eval set, spoofed data is generated through combinations of unseen spoofing algorithms similar to those used to synthesize in the development set.

In the eval set of LA, 13 spoofed systems are included comprising of 7 text-to-speech syntheses, i.e., A07-A12, A16, 3 TTS-VC systems, i.e., A13, A14, A15, and 3 VC spoofed systems, such as A17-A19 that are used to generate the spoof speeches. We employed an experiment based on three phases to assess the effectiveness of the *DeepDet* for VC and TTS systems.

In the Logical Access (LA) scenario, the training set comprises 2580 bonafide utterances and 22,800 spoofed utterances. The development set consists of 2548 bonafide and 22,296 spoofed utterances, while the evaluation set includes 7355 bonafide and 63,882 spoofed utterances.

More precisely, first, we utilized the spoofed and bonafide samples of TTS from a train set of logical access datasets for the training of our DL model. The results are shown in Table 4. We attained an EER of 0.50% and t-DCF of 0.005. Second, we utilized samples from the train set of the VC system of the LA dataset for the training of our proposed model to analyze the performance. We attained an EER of 0.90% and t-DCF of 0.06. It is concluded from the results that the *DeepDet* performs better for the detection of TTS spoofed speeches than VC spoofed detection. The reason behind the better performance of *DeepDet* for TTS spoofing detection is that the voice generated from the VC systems is based on the original audio samples' periodic characteristics. However, TTS systems lack this property. Third, we performed an experiment using the general LA dataset to analyze the performance of the proposed model and achieved 0.042 EER. The overall performance of the proposed system is significant on the LA set. Therefore, we can say that our model, i.e., *DeepDet,* effectively detects the fake audio. Similarly, the experiments are performed for the dev set as well, and the results are reported in Table 4.

5 Ablation study

In this experiment, we analyzed the performance of our proposed DeepDet using varying schemes. First, we assessed the results with the original YAMNet. Then, we attached the BAM module with YAMNet without

Table 5 Comparison with a base model

Model	Accuracy(%)	Precision(%)	Recall(%)	EER(%)
Base network	90.1	93.4	91.1	2.3
YAMNet with BN and BAN	95.4	97.3	95.4	1.30
DeepDet	99.92	99.2	99.76	0.042

Table 4 Results for synthesized speech and voice conversion

Set category	Spoofing system	Accuracy (%)	Precision (%)	Recall (%)	EER (%)	Min-tDCF
Eval	VC	93.3	97	95	0.90	0.06
	TTS	99.8	99.6	99.4	0.50	0.005
	LA(overall)	99.92	99.2	99.76	0.042	0.0015
Dev	VC	94.1	95	95	0.50	0.11
	TTS	98.8	98.6	99.1	0.010	0.015
	LA(overall)	98.9	98.2	98.9	0.015	0.002

Table 6 Results on PA set of ASVSpoof 2019

Set Category	Accuracy (%)	Precision (%)	Recall (%)	EER (%)	Min-tDCF
Dev	99.65	98.4	98.9	3.11	0.05
Eval	99.8	99.8	99.1	0.43	0.0021

Table 7 Comparison with existing spoofing detection systems

Model	Logical access	
	Dev	Eval
FFT-L-SENet [70]	-	1.14
Attention-based CNN [71]	0.16	1.87
LFCC-PC-DARTS [72]	0.002	4.87
RAWNet2 [73]	-	1.12
W2V-Siamese [74]	0.004	1.15
W2V2-light-DARTS [75]	0.02	1.08
EDL-Det [76]	0.80	0.045
DeepDet	0.015	0.042

changing the BN layers. In the end, we changed the BN layers with IN layers to improve the performance further. The results are reported in Table 5. It is clearly visible from the results that DeepDet attains more remarkable results than the first two schemes.

6 Performance analysis over physical access attacks

In this experiment, we aim to examine the performance of our spoofing audio detector using physical access attacks. Therefore, we transformed the auditory samples of the PA set into mel-spectrograms and passed them to an improved YAMNet's base network, i.e., customized MobileNet for the classification into bonafide and replay samples. We achieved an EER of 0.43% and 3.11% for eval and dev sets. Moreover, the min-tDCF of 0.0021 and 0.05 is achieved for eval and dev sets, as reported in Table 6. The results show that our suggested spoofing detector attained significant performance compared to the existing models. Particularly, our proposed model is based on an improved MobileNet, which utilizes depth-wise separable convolutional layers to extract the most representative features from the mel-spectrograms generated from audio. Therefore, we attained EER, which is less than the EER achieved for the existing system on the eval set, such as in [69]. We believe, after the experiment, that our improved MobileNet is capable of effectively extracting features from replay samples to detect physical access attacks.

7 Performance comparison with existing techniques

In this experiment, we compared *DeepDet* with the existing models for voice spoofing detection. The comparative results are reported in Table 7, which considers the evaluation and development set of the ASVspoof 2019 LA corpus. It can be seen that our proposed spoofing detector attains the lowest EER as 0.0015 and 0.042 for dev and eval sets, outperforming the existing systems. The second lowest EER 0.045 is achieved by EDL-Det, and then W2V2-light-DARTS attains, for the eval set as 1.08. Moreover, the second lowest EER for the dev set is achieved by LFCC-PC-DARTS as 0.02. However, the system attained the highest EER for the eval set, which was 4.87. From this analysis, it is concluded that our spoofing detector can effectively detect various spoofed attacks and voices based on cloning algorithms. More precisely, our proposed algorithm outperforms the existing techniques.

8 Conclusion

In this paper, we have presented a voice spoofing detector, i.e., *DeepDet*, employing an improved deep learning model, YAMNet, to detect synthetic attacks. We employed an improved MobileNet along with the BAM attention module as the base network for feature extraction and classification of mel-spectrograms into bonafide and spoofed samples. An improved MobileNet with BAM effectively captures the sample dynamics, artifacts of cloning algorithms and environment, and microphone variations of the replay attacks. Moreover, the significance of utilizing MobileNet lies in an implication of linearly separable depth convolutional layers that makes it light-weight. The BAM module guides the overall network for extraction of key features from mel-spectrograms. We assessed the performance of the proposed model using a diverse and large-scale dataset, i.e., ASVspoof 2019 corpus, and it was concluded that our system is applicable for the detection of several types of spoofing attacks. More precisely, our model attained an EER of 0.43% and 0.042% for PA and LA attacks correspondingly. Our system effectively distinguishes the various cloning algorithms employed for the generation of speech. Additionally, our comparative assessment with existing models unveils that *DeepDet* outperforms them for various forms of speech spoofing detection, such as cloning-based, text-to-speech, and replay attacks. Furthermore, it is worth mentioning that evaluation samples of the dataset include speeches from unseen speakers, and our proposed system attained excellent results on the ASVspoof 2019 evaluation set. Therefore, we believe that DeepDet is a robust spoofing detector due to its effectiveness on the evaluation set of ASVspoof 2019. In the future, we aim to cross-validate our model on other voice spoofing datasets as well and further improve the performance.

Code availability

The code can be provided on demand.

Authors' contributions

All authors contributed to the study's conception and design.

Funding

The authors extend their appreciation to the Deputyship for Research and Innovation, "Ministry of Education" in Saudi Arabia for funding this research (IFKSUOR3–561–2).

Availability of data and materials

The datasets generated and analyzed during the current study are available in the [asvspoof.org], [https://www.asvspoof.org/database].

Declarations

Ethics approval and consent to participate

All authors gave their consent to participate.

Consent for publication

All authors agreed to submit the manuscript for possible publication in the journal.

Competing interests

The authors declare that they have no competing interests.

Received: 15 September 2023 Accepted: 20 February 2024
Published online: 01 April 2024

References

1. M. Toğaçar, Using DarkNet models and metaheuristic optimization methods together to detect weeds growing along with seedlings. Eco. Inform. **68**, 101519 (2022)
2. S. Korse et al., *PostGAN: A GAN-Based Post-Processor to Enhance the Quality of Coded Speech* (2022). arXiv preprint arXiv:2201.13093
3. S.S. Shah et al., Prosodic speech synthesis of narratives depicting emotional diversity using deep learning, in *advanced computational paradigms and hybrid intelligent computing*. (Springer, 2022), pp.31–42
4. V. García, I. Hernáez, E. Navas, Evaluation of tacotron based synthesizers for Spanish and Basque. Appl. Sci. **12**(3), 1686 (2022)
5. T. Okamoto et al., Neural speech-rate conversion with multispeaker WaveNet vocoder. Speech Commun. **138**, 1–12 (2022)
6. Hartmann, K. and K. Giles. *The next generation of cyber-enabled information warfare*. In *2020, the 12th International Conference on Cyber Conflict (CyCon)* 2020. IEEE.
7. Y. Mirsky, W. Lee, The creation and detection of deepfakes: a survey. ACM Computing Surveys (CSUR) **54**(1), 1–41 (2021)
8. National Academies of Sciences, E., and Medicine, *Implications of artificial intelligence for cybersecurity: Proceedings of a workshop*. 2020: National Academies Press.
9. Korshunov, P., et al. *Overview of BTAS 2016 speaker anti-spoofing competition*. In *2016, IEEE held the 8th International Conference on Biometrics Theory, applications, and Systems (BTAS)*. 2016. IEEE.
10. Wu, H., et al. *Defense against adversarial attacks on spoofing countermeasures of ASV*. in *ICASSP 2020–2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2020. IEEE.
11. Wu, D. *An audio classification approach based on machine learning*. In the *2019 International Conference on Intelligent Transportation, Big Data & Smart City (ICITBS)*. 2019. IEEE.
12. M. Todisco et al., *ASVspoof 2019: Future horizons in spoofed and fake audio detection* (2019). arXiv preprint arXiv:1904.05441
13. H. Dinkel, Y. Qian, K. Yu, Investigating raw wave deep neural networks for end-to-end speaker spoofing detection. IEEE/ACM Transactions on Audio, Speech, and Language Processing **26**(11), 2002–2014 (2018)
14. Md Sahidullah, Héctor Delgado, Massimiliano Todisco, Andreas Nautsch, Xin Wang, Tomi Kinnunen, Nicholas Evans, Junichi Yamagishi & Kong-Aik Lee Introduction to voice presentation attack detection and recent advances, Advances in Computer Vision and Pattern Recognition book series (ACVPR), 2023.
15. A. Chintha et al., Recurrent convolutional structures for audio spoof and video deepfake detection. IEEE Journal of Selected Topics in Signal Processing **14**(5), 1024–1037 (2020)
16. G. Lavrentyeva et al., *STC antispoofing systems for the ASVspoof2019 challenge* (2019). arXiv preprint arXiv:1904.05576
17. He, K., et al. *Identity mappings in deep residual networks*. In *European conference on computer vision*. 2016. Springer.
18. M. Alzantot, Z. Wang, M.B. Srivastava, *Deep residual neural networks for audio spoofing detection* (2019). arXiv preprint arXiv:1907.00501
19. C.-I. Lai et al., *ASSERT: Anti-spoofing with squeeze-excitation and residual networks* (2019). arXiv preprint arXiv:1904.01120
20. Monteiro, J., J. Alam, and T.H. Falk. *An ensemble-based approach for generalized detection of spoofing attacks to automatic speaker recognizers*. In *ICASSP 2020–2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2020. IEEE.
21. N.K. Verma et al., Intelligent condition-based monitoring using acoustic signals for air compressors. IEEE Trans. Reliab. **65**(1), 291–309 (2015)
22. L. Nanni et al., Combining visual and acoustic features for audio classification tasks. Pattern Recogn. Lett. **88**, 49–56 (2017)
23. L. Lu, H.-J. Zhang, H. Jiang, Content analysis for audio classification and segmentation. IEEE Transactions on speech and audio processing **10**(7), 504–516 (2002)
24. J. Zhao, X. Mao, L. Chen, Speech emotion recognition using deep 1D & 2D CNN LSTM networks. Biomed. Signal Process. Control **47**, 312–323 (2019)
25. Carey, M.J., E.S. Parris, and H. Lloyd-Thomas. *A comparison of features for speech music discrimination*. In *1999, the IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings. ICASSP99 (Cat. No. 99CH36258)*. 1999. IEEE.
26. Stylianou, Y. *Voice transformation: a survey*. In *2009 IEEE International Conference on Acoustics, Speech, and Signal Processing*. 2009. IEEE.
27. Z. Wu et al., Spoofing and countermeasures for speaker verification: a survey. Speech Commun. **66**, 130–153 (2015)
28. Z. Wu et al., Anti-spoofing for text-independent speaker verification: an initial database, comparison of countermeasures, and human performance. IEEE/ACM Transactions on Audio, Speech, and Language Processing **24**(4), 768–783 (2016)
29. Y.-H. Chao et al., Using kernel discriminant analysis to improve the characterization of the alternative hypothesis for speaker verification. IEEE Trans. Audio Speech Lang. Process. **16**(8), 1675–1684 (2008)
30. Ze, H., A. Senior, and M. Schuster. *Statistical parametric speech synthesis using deep neural networks*. In *2013, an international conference on acoustics, speech, and signal processing took place*. 2013. IEEE.
31. Dörfler, M., R. Bammer, and T. Grill. *Inside the spectrogram: convolutional neural networks in audio processing*. In *2017, there was an international conference on sampling theory and applications (SampTA)*. 2017. IEEE.
32. B. Balamurali et al., Toward robust audio spoofing detection: A detailed comparison of traditional and learned features. IEEE Access **7**, 84229–84241 (2019)
33. Y.-H. Chao, Using LR-based discriminant kernel methods with applications to speaker verification. Speech Commun. **57**, 76–86 (2014)
34. S. Yaman, J. Pelecanos, Using polynomial kernel support vector machines for speaker verification. IEEE Signal Process. Lett. **20**(9), 901–904 (2013)
35. R. Loughran et al., Feature selection for speaker verification using genetic programming. Evol. Intel. **10**(1), 1–21 (2017)
36. H. Zhao, H. Malik, Audio recording location identification using acoustic environment signature. IEEE Trans. Inf. Forensics Secur. **8**(11), 1746–1759 (2013)
37. H. Yu et al., Spoofing detection in automatic speaker verification systems using DNN classifiers and dynamic acoustic features. IEEE transactions on neural networks and learning systems **29**(10), 4633–4644 (2017)

38. A. Maccagno et al., A CNN approach for audio classification in construction sites, in *Progresses in Artificial Intelligence and Neural Systems*. (Springer, 2021), pp.371–381
39. S. Bai, J.Z. Kolter, V. Koltun, *An empirical evaluation of generic convolutional and recurrent networks for sequence modeling* (2018). arXiv preprint arXiv:1803.01271
40. C. Zhang, C. Yu, J.H. Hansen, An investigation of deep-learning frameworks for speaker verification antispoofing. IEEE Journal of Selected Topics in Signal Processing **11**(4), 684–694 (2017)
41. D. Paul, M. Pal, G. Saha, Spectral features for synthetic speech detection. IEEE journal of selected topics in signal processing **11**(4), 605–617 (2017)
42. T. Kinnunen et al., *The ASVspoof 2017 challenge: Assessing the limits of replay spoofing attack detection* (2017)
43. AlBadawy, E.A., S. Lyu, and H. Farid. *Detecting AI-synthesized speech using bispectral analysis*. in *CVPR Workshops*. 2019.
44. Luo, A., et al. *A capsule network-based approach for detection of audio spoofing attacks*. In *ICASSP 2021–2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2021. IEEE.
45. Wang, R., et al. *Deepsonar: Towards effective and robust detection of ai-synthesized fake voices*. In *Proceedings of the 28th ACM International Conference on Multimedia*. 2020.
46. H. Dhamyal et al., Fake Audio Detection in Resource-constrained Settings using Microfeatures. Proc. Interspeech **2021**, 4149–4153 (2021)
47. Lea, C., et al. *Temporal convolutional networks: a unified approach to action segmentation*. in *European Conference on Computer Vision*. 2016. Springer.
48. Arık, S.Ö., et al. *Deep voice: Real-time neural text-to-speech*. In *International Conference on Machine Learning*. 2017. PMLR.
49. W. Ping et al., *Deep Voice 3: 2000-Speaker Neural Text-to-Speech* (2017)
50. Tak, Hemlata, Jose Patino, Massimiliano Todisco, Andreas Nautsch, Nicholas Evans, and Anthony Larcher. End-to-end anti-spoofing with rawnet2. In ICASSP 2021–2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 6369–6373. IEEE, 2021.pr
51. Priyanka Gupta, PK Chodingala, HA Patil "Replay spoof detection using energy separation based instantaneous frequency estimation from quadrature and in-phase components", Computer Speech & Language 77, 2023.
52. C. Wang, J. Yi, J. Tao, C. Zhang, S. Zhang, Fu. Ruibo, X. Chen, *TO-Rawnet: improving RawNet with TCN and orthogonal regularization for fake audio detection* (2023). arXiv preprint arXiv:2305.13701
53. Yang, S., et al. *From facial parts responses to face detection: a deep learning approach*. In *Proceedings of the IEEE international conference on computer vision*. 2015.
54. Ng, H.-W., et al. *Deep learning for emotion recognition on small datasets using transfer learning*. In *Proceedings of the 2015 ACM on international conference on multimodal interaction*. 2015.
55. A.G. Howard et al., *Mobilenets: efficient convolutional neural networks for mobile vision applications* (2017). arXiv preprint arXiv:1704.04861
56. M. Plakal and D. Ellis, Y., " . [Online]. Available: https://github.com/tensorflow/models/tree/master/research/audioset/yamnet, *[Online]. Available:* https://github.com/tensorflow/models/tree/master/research/audioset/yamnet, Jan 2020.
57. N.S. Keskar et al., *On large-batch training for deep learning: generalization gap and sharp minima* (2016). arXiv preprint arXiv:1609.04836
58. Maas, A.L., A.Y. Hannun, and A.Y. Ng. *Rectifier nonlinearities improve neural network acoustic models*. In *Proc. icml*. 2013. Citeseer.
59. L. Sifre, S. Mallat, *Rigid-motion scattering for texture classification* (2014). arXiv preprint arXiv:1403.1687
60. D. Ulyanov, A. Vedaldi, V. Lempitsky, *Instance normalization: the missing ingredient for fast stylization* (2016). arXiv preprint arXiv:1607.08022
61. J. Park et al., *Bam: Bottleneck attention module* (2018). arXiv preprint arXiv:1807.06514
62. Wu, Z., et al. *ASVspoof 2015: the first automatic speaker verification spoofing and countermeasures challenge*. In *Sixteenth annual conference of the International Speech Communication Association*. 2015.
63. X. Wang et al., ASVspoof 2019: A large-scale public database of synthesized, converted and replayed speech. Comput. Speech Lang. **64**, 101114 (2020)
64. C. Veaux, J. Yamagishi, K. MacDonald, *Superseded-cstr vctk corpus: English multi-speaker corpus for cstr voice cloning toolkit* (2016)
65. M. Morise, F. Yokomori, K. Ozawa, World: a vocoder-based high-quality speech synthesis system for real-time applications. IEICE Trans. Inf. Syst. **99**(7), 1877–1884 (2016)
66. Oord, A.v.d., et al., *Wavenet: a generative model for raw audio*. arXiv preprint arXiv:1609.03499, 2016.
67. D. Griffin, J. Lim, Signal estimation from modified short-time Fourier transform. IEEE Trans. Acoust. Speech Signal Process. **32**(2), 236–243 (1984)
68. Wang, X., S. Takaki, and J. Yamagishi. *Neural source-filter-based waveform model for statistical parametric speech synthesis*. in *ICASSP 2019–2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2019. IEEE.
69. Tanaka, K., et al. *Synthetic-to-natural speech waveform conversion using cycle-consistent adversarial networks*. in *2018 IEEE Spoken Language Technology Workshop (SLT)*. 2018. IEEE.
70. M. Todisco, H. Delgado, N. Evans, Constant Q cepstral coefficients: A spoofing countermeasure for automatic speaker verification. Comput. Speech Lang. **45**, 516–535 (2017)
71. Y. Zhang, W. Wang, P. Zhang, *The effect of silence and dual-band fusion in anti-spoofing system* (2021)
72. Ling, H., et al. *Attention-based convolutional neural network for ASV spoofing detection*. in *Interspeech*. 2021.
73. S. Liu et al., Recent progress in the CUHK dysarthric speech recognition system. IEEE/ACM Transactions on Audio, Speech, and Language Processing **29**, 2267–2281 (2021)
74. Tak, H., et al. *End-to-end anti-spoofing with rawnet2*. in *ICASSP 2021–2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2021. IEEE.
75. Xie, Y., Z. Zhang, and Y. Yang. *Siamese Network with wav2vec Feature for Spoofing Speech Detection*. in *Interspeech*. 2021.
76. Wang, C., et al. *Fully automated end-to-end fake audio detection*. in *Proceedings of the 1st International Workshop on Deepfake Detection for Audio Multimedia*. 2022.
77. R. Mahum, A. Irtaza and A. Javed, EDL-Det: A robust TTS detector using VGG19-based YAMNet and Ensemble Learning Block, in IEEE Access, https://doi.org/10.1109/ACCESS.2023.3332561.

Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.