applied sciences

MDPI

Article

Attention-Based LSTM Algorithm for Audio Replay Detection in Noisy Environments

Jiakang Li, Xiongwei Zhang *, Meng Sun *, Xia Zou and Changyan Zheng

Lab of Intelligent Information Processing, Army Engineering University, Nanjing 210007, China; jkangli@163.com (J.L.); xiazou@163.com (X.Z.); echoaimaomao@163.com (C.Z.)

* Correspondence: xwzhang9898@163.com (X.Z.); sunmengccjs@gmail.com (M.S.)

Received: 18 March 2019; Accepted: 10 April 2019; Published: 13 April 2019

Featured Application: In this paper, the proposed attention-based long short-term memory (LSTM) algorithm and bagging methods accurately detected the replay spoofing attacks in noisy environments. For example, in voice authentication of bank-accounts or mobile devices, an attacker could be effectively prevented from replaying the voice of a legitimate user by using this algorithm. This algorithm has potential important applications in biometrics and information security, especially for improving the security of the automatic speaker verification systems in noisy environments.

Abstract: Even though audio replay detection has improved in recent years, its performance is known to severely deteriorate with the existence of strong background noises. Given the fact that different frames of an utterance have different impacts on the performance of spoofing detection, this paper introduces attention-based long short-term memory (LSTM) to extract representative frames for spoofing detection in noisy environments. With this attention mechanism, the specific and representative frame-level features will be automatically selected by adjusting their weights in the framework of attention-based LSTM. The experiments, conducted using the ASVspoof 2017 dataset version 2.0, show that the equal error rate (EER) of the proposed approach was about 13% lower than the constant Q cepstral coefficients-Gaussian mixture model (CQCC-GMM) baseline in noisy environments with four different signal-to-noise ratios (SNR). Meanwhile, the proposed algorithm also improved the performance of traditional LSTM on audio replay detection systems in noisy environments. Experiments using bagging with different frame lengths were also conducted to further improve the proposed approach.

Keywords: audio replay attack; noise robustness; attention mechanism; long short-term memory

1. Introduction

With the growing popularity of automatic speaker verification (ASV) systems, the attacks on them have posed significant security threats, which requires that the ASV systems have the ability to detect spoofing attacks. Generally, spoofing attacks can be categorized into four types: impersonation, synthesis, conversion, and replay [1]. In order to promote research on anti-spoofing, the automatic speaker verification spoofing and countermeasures (ASVspoof) challenge [2] was first launched in 2015, which focused on discriminating between synthesized or converted voices and those uttered by a human. The second challenge, i.e., the ASVspoof 2017 challenge [3], focused on detecting replay spoofing to discriminate whether a given speech was the voice of an in-person human or the replay of a recorded speech.

Replay attack refers to when an attacker uses a high-fidelity recording device to record the voice of a legitimate authentication system user and then uses the recorded playback through the device on the

ASV system, thereby achieving an attack behavior [4]. With the development of electronic technology, the performance of high-fidelity recording and playback equipment has been considerably improved. A replay attack is not difficult to implement and does not require any specialized knowledge, which poses a significant threat to ASV systems [5]. Therefore, the research on replay attack detection has become increasingly important. Villalba et al. [6] used far-field replay speech recordings to investigate the vulnerability of an ASV system and showed that the equal error rate (EER) of the ASV system increased from 1% to nearly 70% when the human voice was recorded directly and replayed to the device. Algre et al. [5] further evaluated the risk of replay attack by using the 2005 and 2006 National Institute of Standards and Technology Speaker Recognition Evaluation dataset (NIST' 05 and NIST' 06) corpus and six kinds of ASV systems. Their results showed that the low-effort replay attack posed a significant risk to all the ASV systems tested. However, the work presented above only reported the results in conditions without background noise, thus it is still not clear how background noise affects the performance of spoofing detection. Therefore, in order to enhance the security of ASV systems, spoofing detection in noisy environment is investigated in this paper.

2. Related Work

Over the years, different methods have been tested to explore spoofing detection. A replay-detection algorithm, based on peakmap audio features, was proposed in [4], which calculated the similarity of the peakmap features between the recorded test utterance and the original enrollment one. If the similarity was above a certain threshold, the utterance was determined to be a replay attack. The replay-detection performance was further improved by adopting a relative similarity score in [7].

Zhang et al. used Mel frequency cepstral coefficients (MFCCs) to detect replay [8]. They believed that the silent segments of a speech would better detect the channel source than the spoken segments. Therefore, voice activation detection (VAD) was utilized to extract the silent segments of a speech and a universal background model (UBM) was created to model the difference between the speaker's testing utterance and their enrollment utterance. However, the use of acoustic signal processing to detect replay attacks is considered to be challenging due to the unpredictable changes in voice recording environments, recording equipment, etc. [3]. For example, artifacts introduced by the acoustic surroundings, such as reverberation, might be confused with the artifacts introduced by playback in some cases. Li et al. [9] tried to use machine learning to detect replay attacks, but only obtained poor performance due to overfitting.

In the ASVspoof 2017 challenge, an official corpus for detecting replay attack and a baseline system, based on the Gaussian mixture model (GMM) with constant Q cepstral coefficient (CQCC) features [10], were provided. The challenge required participants to propose a method that distinguished between genuine speech and a replay recording, where a total of 49 submissions were received from the participants. The average EER of all the submissions was 26.01% [3], where only 20 out of 49 submissions obtained lower EERs than the GMM and CQCC baseline. Among all of the submissions, the best detection performance was reached when using deep convolutional neural networks on spectrograms [11]. This method used an ensemble of three techniques: convolutional neural network (CNN) with recurrent neural network (RNN), light CNN (LCNN), and support vector machine (SVM) i-vector. In addition, Patil et al. [12] proposed variable length Teager energy operator-energy separation algorithm-instantaneous frequency cosine coefficients (VESA-IFCC), and Li et al. [9] utilized a deep neural network (DNN) model with L-Fbanks cepstral coefficient (LFCC) features to identify the genuine and the spoofing speech. All these algorithms achieved better results when compared to the other submissions on the dataset.

The work in this paper is based on the official corpus of the ASVspoof 2017 challenge, i.e., ASVspoof2017 dataset Version 2.0. However, the genuine utterances of the dataset were collected in clean environments (i.e., without any background noises), while the spoof utterances were recorded in environments with various background noises. The genuine utterance setting conflicts with real-world scenarios with complex, noisy environments where the ASV system typically works. In fact,

genuine utterances may have background noises, while spoof utterances will contain both the genuine utterances' noises and the noises of the recording environment. Therefore, noisy environments will definitely create challenges in distinguishing between genuine and spoof utterances. Therefore, it is worth exploring the effectiveness of a noise-robust algorithm on spoofing detection, which is the motivation of this paper.

3. Motivation

The previous related work has confirmed that replay attack is still quite a threat to ASV systems, although a couple of replay detection methods have been proposed with success. Towards noise-robust detection, countermeasures and detection algorithms still have room for improvement, especially in complex, noisy environments. Figure 1 shows two pairs of genuine human voices and their corresponding replay speech from the ASVspoof 2017 dataset Version 2.0. The figure tells us the following:

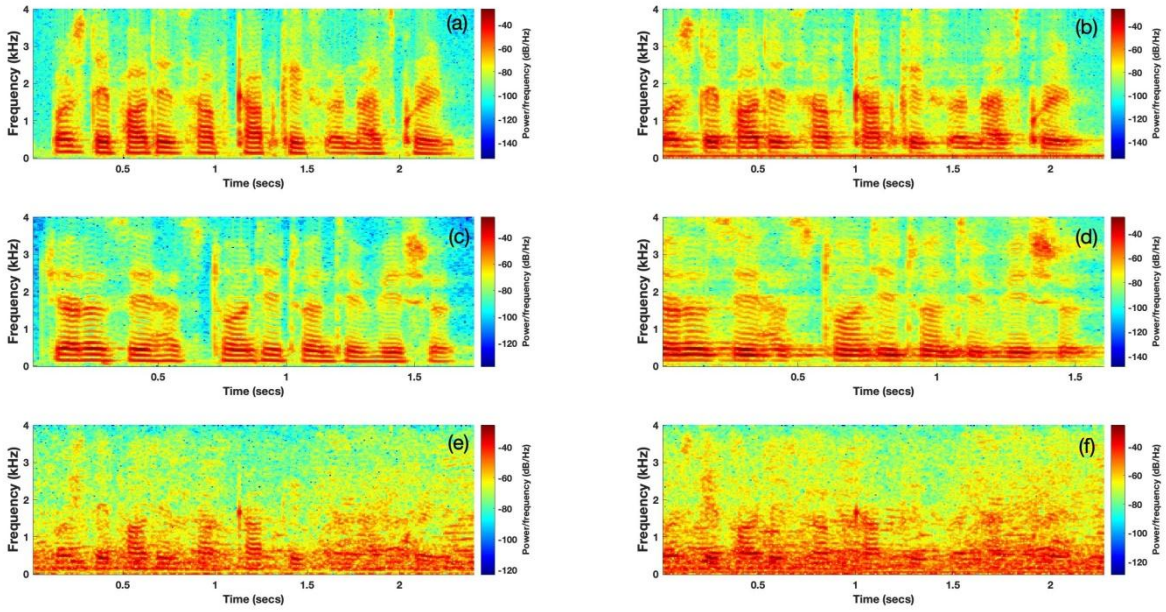

Figure 1. Spectrogram of genuine and replay utterances. (**a**,**c**) are genuine human voices while (**b**,**d**) are replay recordings of (**a**,**c**), respectively. Spectrograms (**e**,**f**) are the noise contaminated versions of (**a**,**b**), respectively, where babble noise with 0 dB signal-to-noise ratio (SNR) was added.

(1) The durations of the utterances are different. Hence, the chosen algorithm should be able to process signals with variable lengths.

(2) The first pair of the genuine (a) and spoof (b) utterances differ greatly in their low frequency regions, while the differences in the second utterance pair, (c) and (d), lie mainly in the high frequency regions. Therefore, distinguishing features can occur over all frequency bands and it is, thus, not appropriate to focus on only high or low frequency bands.

(3) (e) and (f) are the noise-contaminated utterances of (a) and (b), respectively. The spectrograms show that the utterances in noisy environments are more complicated than those of the clean environments. For example, when comparing (a) and (e), it is obvious that some of the distinguishing features of (a) have been dispersed when noise occurs. This phenomenon highlights the difficulties in spoofing detection for noisy utterances, as will be numerically explained in Section 5.3.

In this paper, we aim to improve the spoofing detection algorithm by overcoming the problems presented at the beginning of this section, and propose a robust algorithm for replay attack detection in noisy environments. Our research motivations are summarized as follows:

(1) The sequential nature of speech motivated us to choose long short-term memory (LSTM) to obtain a representation of an utterance. By using LSTM, contextual information can be easily transformed into a vector as input for any back-end classifier.

(2) Considering that different audio frames may contribute differently to the task of replay detection, especially under noisy environments where the frames are contaminated to different degrees by noise, an attention mechanism [13] was used to perform automatic weighting of the frames.

(3) When programming RNNs, for example with TensorFlow, utterance inputs should be the same length, which is normally done by zero-padding the short inputs in the batch to make their length meet a prefixed constant. The lengths of the utterances vary significantly in the ASVspoof 2017 dataset, where the longest utterance has 1423 frames while the shortest one has only 86 frames. There would be too many zeros in the short utterances if simply zero-padding the short ones, which would make the data distribution severely biased toward the padded utterances. In fact, given our research on RNNs, LSTMs, and gated recurrent units (GRUs), the sequence length of the model performs as a hyper-parameter which impacts the model's performance. Therefore, it is also worth investigating the impact of the sequence length on the accuracy of replay detection in the ASV spoofing scenario. Hence, the split/padding method was used to obtain a group of speech segments with uniform length, and is presented in Section 5.2.3. For an input sequence whose length was longer than the prefixed one, splitting was utilized to cut the sequence into smaller sections that met the prefixed length. For an input sequence whose length was shorter than the prefixed one, padding was utilized by repeating the sequence until the prefixed length was reached. In this way, fewer zeros are padded onto the sequences, their segments will have a uniform length, and they are ready to be RNN inputs. The split/padding method is detailed in Section 5.2.3.

(4) Given the split/padding approach described above, a sequence may be divided into several segments, each of which was analyzed to decide on whether it is a genuine or spoofing utterance. Therefore, for the original whole input sequence, it was easy to use the bagging method in ensemble learning to combine the decisions derived on its individual segments. Section 5.2.4 explains the bagging method in detail.

The rest of this paper is organized as follows: Section 4 describes the attention-based LSTM algorithm (AB-LSTM) and the network architecture for replay attack detection, experimental settings and results are presented in Section 5, and the final section, Section 6, concludes the paper.

4. The Proposed Algorithm

In this section, the attention mechanism and the AB-LSTM are introduced. The proposed network architecture for replay attack detection is subsequently presented.

4.1. Attention Mechanism

4.1.1. Traditional Attention

Attention mechanisms have been applied in many tasks and have achieved remarkable results, such as speech recognition [14–16], natural language processing [17], and machine translation [18]. An attention mechanism [13,14] can adaptively learn the relationship of the inputs at several time steps and predict the current time step by using a weight vector. For a sequence with state h_j at each step, a super vector c_t can be computed by,

$$c_t = \sum_{j=1}^{T} \alpha_{tj} h_j, \tag{1}$$

where T is the number of steps of the input sequence and α_{tj} denotes the weight vector, computed at step t for each state h_j. The weight vectors, α_{tj}, are then computed by using an intermediate vector, e_{tj},

$$e_{tj} = \varphi\left(s_{t-1}, h_j\right), \tag{2}$$

where s_{t-1} is the output at step $t-1$ and $\varphi(\cdot)$ is a learned function, which performs as an important scalar determined by h_j and s_{t-1}. Then the value of the weight vectors, α_{tj}, are computed by,

$$\alpha_{tj} = \frac{\exp\left(e_{tj}\right)}{\sum_{k=1}^{T} \exp(e_{tk})}. \tag{3}$$

This attention mechanism is able to recursively generate weight vectors given the state of the current step. The frames that contribute the most to the final task tend to have higher weights in the entire super vector, while the frames that contribute less to the final task tend to have lower weights. According to Equations (1) to (3), an utterance is encoded to a super vector c, which acts as an input to a back-end genuine/spoof classifier.

4.1.2. Feed-Forward Attention

In the traditional attention mechanism introduced in Section 4.1.1, $\varphi(\cdot)$ in Equation (2) is a learnable function, which depends on the newly generated output, s_{t-1}, and the current state, h_j. In for feed-forward attention, $\varphi(\cdot)$ only depends on the current state h_t, so Equations (2) and (3) become,

$$e_t = \varphi(h_t), \tag{4}$$

and

$$\alpha_t = \frac{\exp(e_t)}{\sum_{k=1}^{T} \exp(e_k)}, \tag{5}$$

respectively, where α_t represent the value of the weight vectors, e_t and e_j represent the intermediate vectors. Thus, the attention mechanism is actually modeled by a super vector, c, of the input sequence by computing an adaptive weighted average of the state sequence h,

$$\mathrm{c} = \sum\nolimits_{t=1}^{T} \alpha_t h_t, \tag{6}$$

The feed-forward attention mechanism is shown in Figure 2. Intuitively, an attention mechanism will put emphasis on the important steps to yield high values in the vector c when modeling sequences.

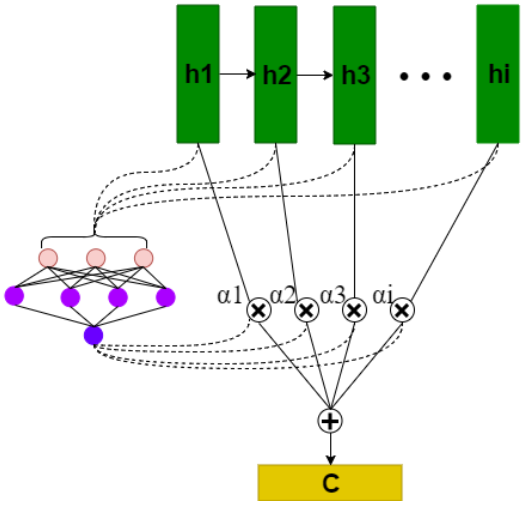

Figure 2. The structure of the "feed-forward" attention mechanism [13,19]. Vectors in the state sequence h_i are fed into the learnable function $\varphi(h_i)$ to produce a weight vector α. The vector c is computed as a weighted average of h_i with the weighting given by the weight vector α.

4.2. Attention-Based LSTM

In this section, for the hidden states of the LSTM output, the feed-forward attention mechanism was utilized to adaptively calculate the weight of every frame. The weights are denoted by $\alpha = \{\alpha_1, \cdots, \alpha_i, \cdots, \alpha_N\}$ for the hidden states, $\mathrm{h} = \{h_1, \cdots, h_i, \cdots, h_N\}$, of frames 1, 2, ..., N, where $\{\alpha_i \geq 0, i = 1, \cdots, N\}$. $h_i^{(j)}$ represents the j-th entry of the frame i in the hidden state h. $w^{(j)}$ represents the trainable parameters. Then the weights, computed by the attention mechanism using sigmoid function, can be defined as,

$$u_i = h_i^{(j)} \cdot w^{(j)}, \tag{7}$$

and

$$\alpha_i = \frac{e^{\frac{1}{1+e^{-ui}}}}{\sum_{i=1}^{N} e^{\frac{1}{1+e^{-ui}}} + \varepsilon}, \tag{8}$$

where the weighted, hidden state, $\widetilde{h}_i$, and the super vector, c, are obtained by,

$$\widetilde{h}_i = \alpha_i \cdot h_i^T, \tag{9}$$

and

$$\mathrm{c} = \sum_{i=1}^{N} \widetilde{h}_i. \tag{10}$$

4.3. Network Architecture for Replay Attack Detection

The proposed framework is shown in Figure 3. The network architecture is as follows: (1) the frame-level features are extracted from the input utterance, (2) the hidden states of the whole sequence are obtained by LSTM, (3) a batch-normalization layer is utilized to reduce overfitting and to improve the model's performance, (4) attentional weighting of the hidden state is transformed into a super vector, and (5) fully-connected layers followed by a softmax layer are used to make a final decision on whether the voice is genuine or a spoof.

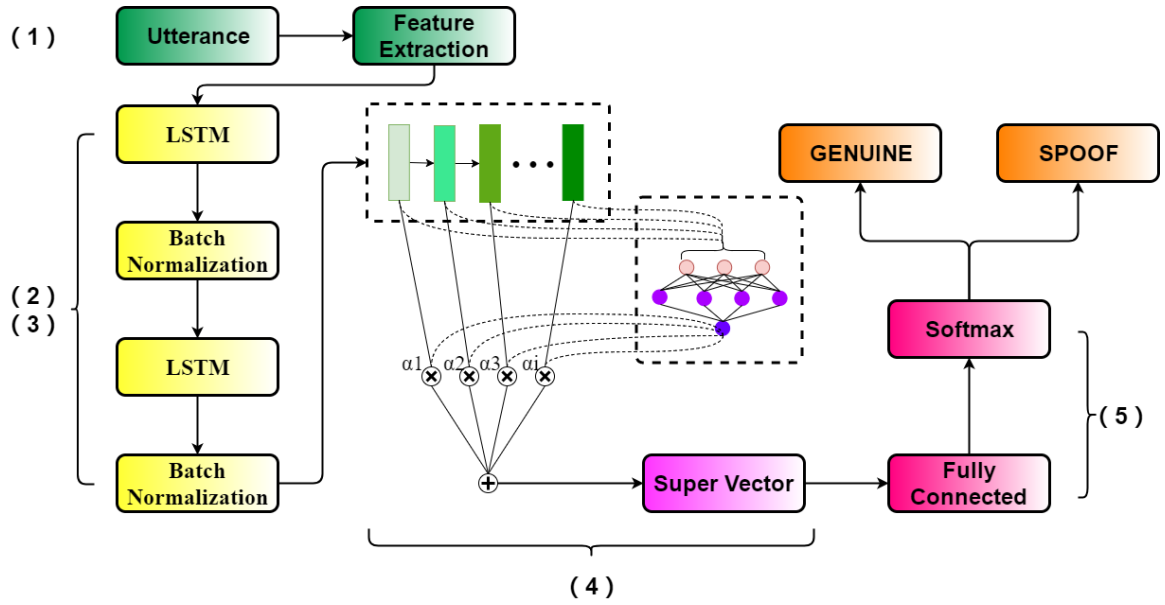

Figure 3. Network architecture for replay attack detection with attention mechanism.

5. Experiments and Results

5.1. ASVspoof 2017 Challenge and Dataset

The ASVspoof 2017 Version 2.0 dataset [3,20] includes three parts: training set, development set, and evaluation set. Each dataset contains both genuine and spoof utterances as shown in Table 1.

Table 1. The detailed information of the ASVspoof 2017 Version 2.0 dataset. The number represents the quantity of utterances in each subset.

Subset	Speakers	Genuine	Spoof
Training	10	1508	1508
Development	8	760	950
Evaluation	24	1298	12008
Total	42	3566	14466

5.2. Experimental Setup

In order to comprehensively evaluate the performance ability of traditional LSTM and AB-LSTM on spoof detection, experiments were conducted in both clean and noisy environments, respectively. Firstly, the CQCC features of each utterance were extracted. Secondly, the split/padding method was used to process the utterance and to obtain segments with the same length. Traditional LSTM and AB-LSTM were then utilized to handle the frame-level features and to decide on genuine or spoof status through bagging methods. Finally, noises with different levels of signal-to-noise ratios (SNRs) were added to the original dataset, and the results on replay detection in noisy environments were obtained by repeating the above steps.

5.2.1. Baseline

The baseline system of the ASVspoof 2017 Challenge used CQCC features [10] with a standard binary GMM classifier for genuine and spoof utterances [3]. For each utterance, the log-likelihood score was obtained from the two GMM classifiers and the final classification decision was based on the log-likelihood ratio. For the experiments in noisy environments, in order to be consistent with the given baseline in the clean background environment [3], we still used CQCC-GMM as the baseline.

5.2.2. Acoustic Features and SNR Settings

In this paper, $D = 90$ dimensional features, with 30 CQCCs, 30 Δ (delta coefficients) and 30 $\Delta\Delta$ (delta-delta coefficients), were utilized. Each frame of an utterance was processed by a 25 ms Hamming window with a 10 ms shift. The number of bins per octave was set to 96. The highest and lowest frequencies were set to 8000 Hz and 16 Hz, respectively. For the experiments in noisy environments, four different kinds of noises were added to the original ASVspoof 2017 Version 2.0 dataset [20]. These four kinds of noises were white, babble, f16, and leopard. The experiments were conducted in four different SNR conditions: -5 dB, 0 dB, 5 dB, and 10 dB.

5.2.3. Split/Padding Method

In the experiments, each utterance had three choices on its prefixed segment length, i.e., 100 frames, 200 frames, and 300 frames. The detailed split/padding methods are shown below. In the padding method (a), for utterances whose length was less than the prefixed value, the front frames were copied and filled to the end of the utterance until reaching the prefixed length. For example, if the specified length was 300 frames, an utterance with 80 frames was extended to 300 frames by repeating its frames, as illustrated in Figure 4.

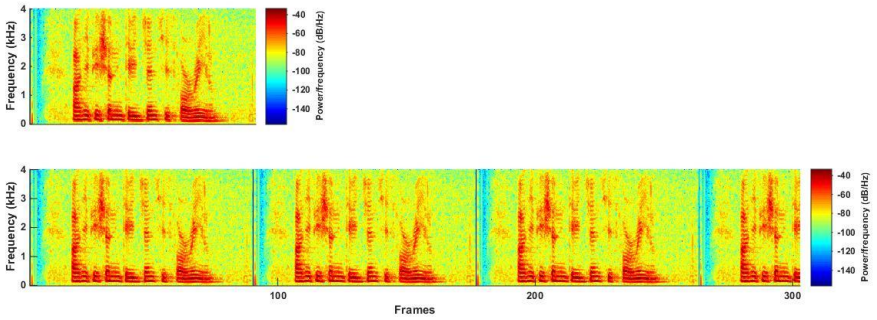

Figure 4. An illustration of the padding method (a).

In the split method (b), for utterances whose length was longer than the prefixed length, the utterance was split. For example, if the prefixed length was 100, an utterance with 250 frames was first extended to 300 frames, according to method (a), and was then be split into three segments, each with 100 frames, as is shown in Figure 5.

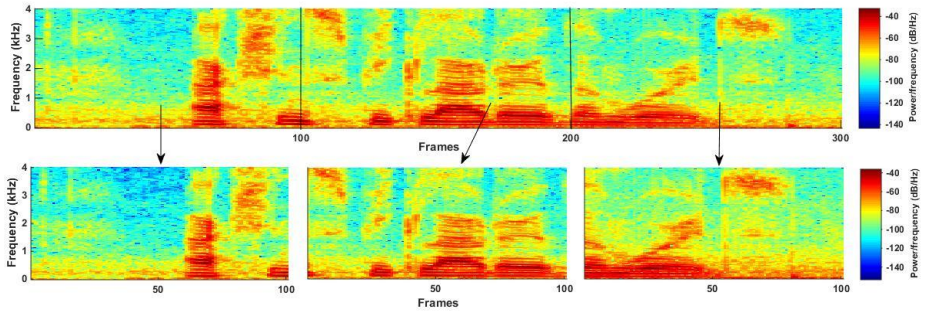

Figure 5. An illustration of the split method (b).

5.2.4. Bagging Methods

By using the split/padding approach presented above, the resulting segments had the same length. A relatively long utterance was, thus, divided into several segments. For each segment, the spoof detection algorithm decided on utterance type. To assess the whole utterance, a bagging approach was used to combine the results of the segments of this utterance. The bagging method was accomplished using two steps (Figure 6). The first step was bagging the decisions derived for each of the segments. That is,

$$\widetilde{F}_a = \frac{\sum_{i=1}^{n} score_i}{n}, \tag{11}$$

where $score_i$ is the classification probabilities of each segment from the utterance, n is the number of segments, and $\widetilde{F}_a$ is the bagging score. The second step was bagging the decisions on the four different kinds of noises, $\widetilde{F}_a$,

$$\widetilde{F}_b = \frac{\sum_{i=1}^{m} \widetilde{F}_{ai}}{m}, \tag{12}$$

where m denotes the number of noises. In the experiments of this paper, m = 4.

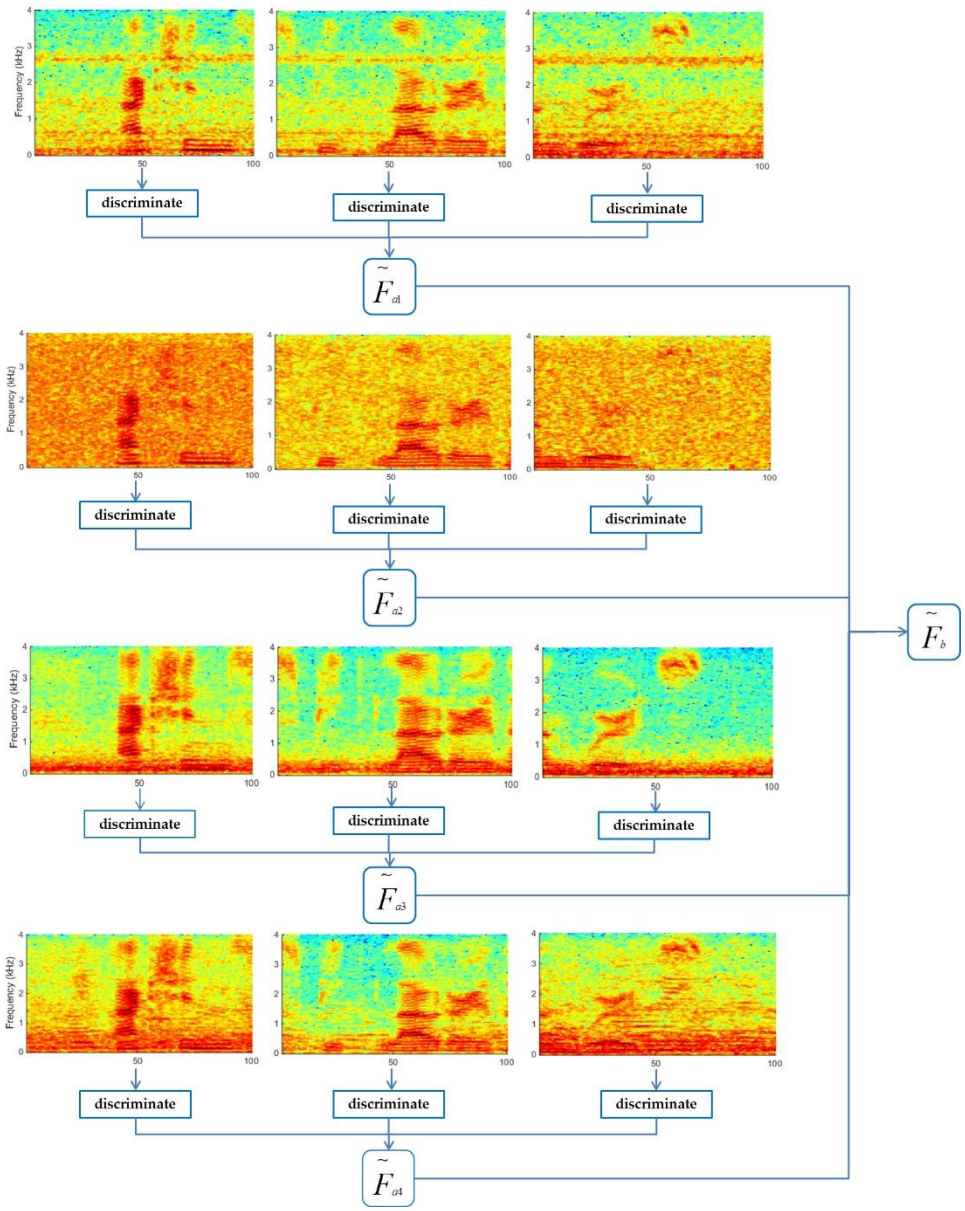

Figure 6. An illustration of the bagging approach used. $\widetilde{F}_{a1}$ denotes the bagging result of the three segments from the same utterance. $\widetilde{F}_{a1}$ to $\widetilde{F}_{a4}$ denote bagging results after adding four different kinds of noises to the original utterance.

5.2.5. Evaluation Metrics

In order to evaluate the performance of the proposed AB-LSTM, contrasting experiments were conducted on AB-LSTM and traditional LSTM, at various SNR levels and for utterances with various lengths. As a classic evaluation metric, EER was used to compare different algorithms. EER is the value where the false alarm rate (FAR) is equal to the false reject rate (FRR). False alarm is defined as the situation where the system incorrectly verifies or identifies a non-target speaker. FAR is the number of false alarms divided by the number of total target trials. False reject is defined as the false rejection of a true target speaker. The FRR is the number of false rejections divided by the total of non-target trials [21]. The CQCC-GMM is adopted from the official MATLAB code in [22]. EER was calculated using the MSR Identity Toolkit Version 1.0 [23].

5.3. Results and Discussion

5.3.1. Experiments in Clean Environments

In this part of the experiment, we used three different frame length segments (100, 200, and 300) as inputs of the network, and calculated the EERs of detection. The architecture of AB-LSTM utilized in the experiments consisted of five recurrent layers with a size of $128 \times 256 \times 256 \times 256 \times 128$. The size of the attention layer was the same as the dimensions of the CQCC features. The last two layers of the model were fully-connected layers with a size of 256×256. The total number of parameters in the network was 1.86 million. As for the traditional LSTM, except for the absence of the attention layer, the remaining parts were the same as those in AB-LSTM. The loss functions of these two methods are binary cross-entropy,

$$\text{loss} = -\sum_{i=1}^{n} \hat{y}_i \log y_i + (1 - \hat{y}_i) \log(1 - \hat{y}_i), \tag{13}$$

where n is the number of samples, y_i is the true category, and $\hat{y}_i$ is the predicted classification result.

After processing all the utterances from the three segment length subsets, using the split/padding method, we used the resultant segments from the training set to train the model, and then used the trained model to perform classification on the development and evaluation subsets.

The detailed EERs of traditional LSTM, AB-LSTM, and other baseline methods from the ASVspoof 2017 challenge [3] are presented in Table 2. As can be seen from the experiments under clean background conditions, the AB-LSTM achieved an EER of 16.86% in the evaluation subset, which ranked among the top submissions of the ASVspoof 2017 challenge [3]. As shown in Table 2, AB-LSTM performed better than traditional LSTM, where the EER of AB-LSTM was reduced by 17.0% when compared to traditional LSTM for the cases with 100 frames. For both traditional LSTM and AB-LSTM, the best EER resulted from the settings with 100 frames.

Table 2. Performance of our algorithm and other algorithms used in the ASVspoof 2017 challenge. Results are in terms of the replay/non-replay EER (%).

Methods		*dev* EER	*eval* EER
LCNN_{FFT}, $\text{SVM}_{\text{i-vect}}$, CNN_{FFT}+RNN [11]		3.95	6.73
VESA-IFCC+CFCCIF [12]		0.12	18.33
LCNN [24]		6.47	16.08
Evolving RNN [25]		18.70	18.20
Traditional LSTM	Frame = 100	**13.24**	**20.32**
	Frame = 200	14.18	21.59
	Frame = 300	13.83	20.97
	Average	13.75	20.96
AB-LSTM	Frame = 100	**9.75**	**16.86**
	Frame = 200	10.69	17.69
	Frame = 300	11.94	18.84
	Average	10.79	17.80
CQCC-GMM (baseline)		12.08	29.35

Figure 7 shows the loss curves of traditional LSTM and AB-LSTM methods observed during training. We can see that the decrease in the loss values was faster with the attention mechanism than without it. Although AB-LSTM had an additional attention layer, the convergence speed of the whole model was significantly higher than that of traditional LSTM. This demonstrated that the attention layer accelerated the optimization of the model by making the network concentrate on frames necessary for genuine and spoof utterance detection.

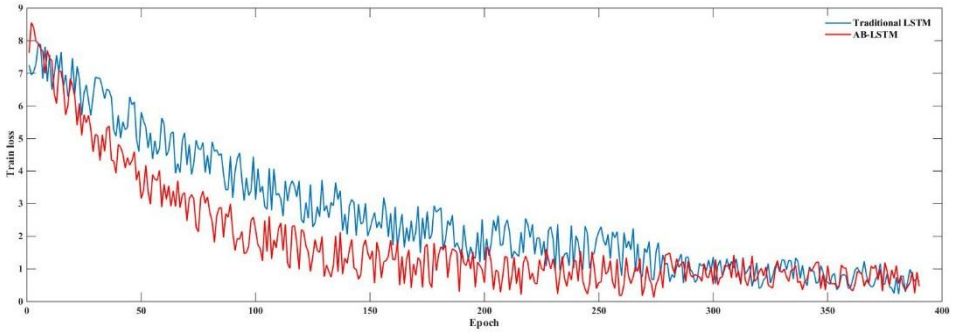

Figure 7. Loss curves of traditional long short-term memory (LSTM) and attention-based LSTM (AB-LSTM).

In order to explore the influence of the network's architecture, another group of experiments were conducted using the 100 frame length segments, combined with the split/padding and bagging methods. All three networks were trained on the training set and evaluated using the evaluation set. Detailed experimental results are shown in Table 3.

Table 3. EER (%) results of the different network's architectures in a clean background environment.

Network's Architecture	Traditional LSTM	AB-LSTM
$128 \times 256 \times 256$	25.74	21.95
$128 \times 256 \times 256 \times 256$	22.61	19.03
$128 \times 256 \times 256 \times 256 \times 128$	**20.32**	**16.86**

5.3.2. Experiments in Noisy Environments

Detailed EERs (%) of the CQCC-GMM baseline, traditional LSTM, and AB-LSTM with different frame lengths under different SNR environments are shown in Tables 4–7. The average EERs of the different frame lengths and the EERs of the bagging methods are also included. The average EERs of different SNR environments are shown in Table 8. Furthermore, Figure 8 illustrates the impact of SNR on EER.

Table 4. EER [a] (%) and EER [b] (%) for different methods when $SNR = -5$ dB.

Methods		EER [a] (%)		EER [b] (%)	
		dev	*eval*	*dev*	*eval*
CQCC-GMM		48.53	47.52	-	-
Traditional LSTM	Frame = 100	**20.58**	**37.78**	20.62	38.13
	Frame = 200	20.82	38.09		
	Frame = 300	23.87	39.86		
	Average	21.76	38.58	-	-
AB-LSTM	Frame = 100	**18.00**	**33.17**	18.26	33.31
	Frame = 200	18.42	33.88		
	Frame = 300	19.71	34.64		
	Average	18.71	33.90	-	-

[a] Bagging EER of segments. [b] Bagging EER of different length of frame.

Table 5. EER [a] (%) and EER [b] (%) for different methods when $SNR = 0$ dB.

Methods		EER [a] (%)		EER [b] (%)	
		dev	*eval*	*dev*	*eval*
CQCC-GMM		39.70	43.51	-	-
Traditional LSTM	Frame = 100	25.04	40.63		
	Frame = 200	**21.37**	**38.11**	22.15	38.46
	Frame = 300	24.74	41.24		
	Average	23.71	39.99	-	-
AB-LSTM	Frame = 100	20.68	33.62		
	Frame = 200	**19.20**	**33.17**	19.12	33.09
	Frame = 300	18.35	32.94		
	Average	19.41	33.24	-	-

Table 6. EER [a] (%) and EER [b] (%) for different methods when $SNR = 5$ dB.

Methods		EER [a] (%)		EER [b] (%)	
		dev	*eval*	*dev*	*eval*
CQCC-GMM		45.99	48.95	-	-
Traditional LSTM	Frame = 100	**19.00**	**36.43**		
	Frame = 200	22.18	39.12	20.77	37.56
	Frame = 300	24.66	40.05		
	Average	21.95	38.53	-	-
AB-LSTM	Frame = 100	**18.43**	**33.27**		
	Frame = 200	19.62	34.06	18.83	33.29
	Frame = 300	19.74	34.85		
	Average	19.26	34.06	-	-

Table 7. EER [a] (%) and EER [b] (%) for different methods when $SNR = 10$ dB.

Methods		EER [a] (%)		EER [b] (%)	
		dev	*eval*	*dev*	*eval*
CQCC-GMM		46.71	46.36	-	-
Traditional LSTM	Frame = 100	**22.38**	**37.91**		
	Frame = 200	23.13	38.14	22.63	38.08
	Frame = 300	23.58	38.33		
	Average	23.03	38.13	-	-
AB-LSTM	Frame = 100	**17.36**	**31.62**		
	Frame = 200	17.57	32.21	17.46	32.11
	Frame = 300	18.09	33.45		
	Average	17.67	32.43	-	-

Table 8. Average EER [a] (%) and EER [b] (%) of all four SNR levels.

Methods		EER [a] (%)		EER [b] (%)	
		dev	*eval*	*dev*	*eval*
CQCC-GMM		45.23	46.59	-	-
Traditional LSTM	Frame = 100	**21.75**	**38.19**		
	Frame = 200	21.88	38.37	21.54	38.06
	Frame = 300	24.21	39.87		
	Average	22.61	38.81	-	-
AB-LSTM	Frame = 100	**18.62**	**32.92**		
	Frame = 200	18.70	33.33	18.42	32.95
	Frame = 300	18.97	33.97		
	Average	18.76	33.41	-	-

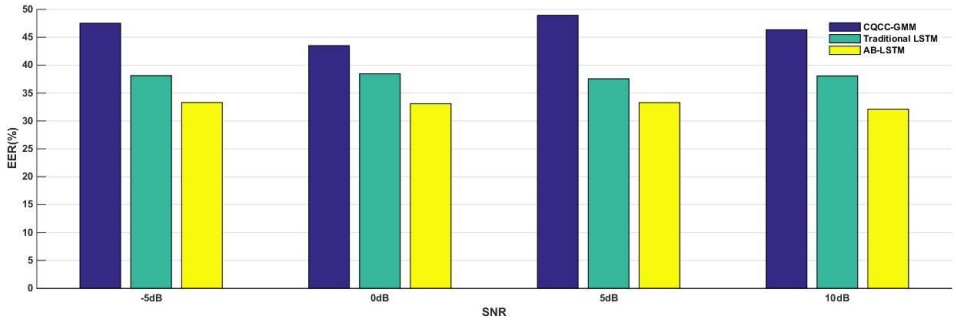

Figure 8. The average EER values of CQCC-GMM, traditional LSTM, and AB-LSTMs at different SNR levels.

Compared with the baseline's EER of 30.60% for the evaluation set in the clean environment [3], the average EER of the four kinds of SNR in noisy environments increased to 45%. This implies that the noise in complex sound environments seriously affected the performance of the baseline system. As for traditional LSTM and the proposed AB-LSTM algorithm, the results of the experiments showed that the AB-LSTM algorithm performed better than the CQCC-GMM baseline for any frame length at any SNR level. It was also seen that the AB-LSTM algorithm achieved better EERs than the traditional LSTM for all parameter settings.

The experiments conducted without the attention mechanism had an average EER of 38.06%, an 8.53% reduction with respect to the baseline system's EER of 46.59%. AB-LSTM obtained an average EER of 32.95% on the evaluation set, which was 13.64% and 5.11% lower than the EER for the baseline system and the traditional LSTM, respectively. The improvement in the EER using the AB-LSTM algorithm, compared to the baseline and traditional LSTM, might be attributed to the attention mechanism, which made the model focus adaptively on the distinguishing utterance information. These results demonstrated the effectiveness of the attention mechanism in replay detection.

The experimental results also demonstrated that 100 was a good choice for the prefixed frame length. The detection methods that used the 100 frame segments performed better than with 200 or 300 frames in all the three SNR environments and for the AB-LSTM algorithm. The experiments also demonstrated that bagging using different lengths of inputs performed better than those using simple averaging. In addition, with the decrease of SNR, the EERs of discrimination also decreased, which indicated that the intensity of the noise affected the accuracy of the replay detection.

As can be seen from Tables 4–7 and Figure 8, EERs did not show a significant decreasing trend with the increase in SNR. The reason for this phenomenon may be that the additional noises caused more serious interference with the original speech signal. However, we found that EERs obtained by AB-LSTM in this study were significantly better than the CQCC-GMM baseline method and traditional

LSTM in noisy environments. The algorithm proposed in this paper deserves further study in replay attack detection.

6. Conclusions

In audio replay detection, it is necessary to pick frames that have important impacts on detection. By introducing an attention mechanism to spoof detection using LSTM, we proposed an algorithm to adaptively highlight the important frames. Moreover, two kinds of bagging methods were proposed and tested, from which accurate and stable performance was observed for spoof detection on noisy speech. Extensive experiments, using the ASVspoof 2017 Version 2.0 dataset contaminated by various noises, demonstrated the effectiveness of the proposed detection methods by achieving a 13.18% reduction in EER when compared to the baseline.

Author Contributions: Conceptualization, J.L., M.S. and X.Z. (Xia Zou); Methodology, J.L., M.S. and C.Z.; investigation, X.Z. (Xiongwei Zhang) and M.S.; data curation, J.L.; writing—original draft preparation, J.L.; writing—review and editing, X.Z. (Xiongwei Zhang) and M.S.

Funding: This research was funded by [the Natural Science Foundation of Jiangsu Province for Excellent Young Scholars] grant number [BK20180080].

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Wu, Z.; Evans, N.; Kinnunen, T.; Yamagishi, J.; Alegre, F.; Li, H. Spoofing and countermeasures for speaker verification: A survey. *Speech Commun.* **2015**, *66*, 130–153. [CrossRef]
2. Wu, Z.; Kinnunen, T.; Evans, N.; Yamagishi, J.; Hanilci, C.; Sahidullah, M.; Sizov, A. ASVspoof 2015: The First Automatic Speaker Verification Spoofing and Countermeasures Challenge. *IEEE J. Sel. Top. Signal Process.* **2017**, *11*, 588–604. [CrossRef]
3. Kinnunen, T.; Sahidullah, M.; Delgado, H.; Todisco, M.; Evans, N.; Yamagishi, J.; Lee, K.A. The ASVspoof 2017 challenge: Assessing the limits of replay spoofing attack detection. In Proceedings of the Annual Conference of the International Speech Communication Association, Stockholm, Sweden, 20–24 August 2017.
4. Shang, W.; Stevenson, M. A playback attack detector for speaker verification systems. In Proceedings of the IEEE International Symposium on Communication, St Julians, Malta, 12–14 March 2008.
5. Alegre, F.; Janicki, A.; Evans, N. Re-assessing the threat of replay spoofing attacks against automatic speaker verification. In Proceedings of the International Conference of the Biometrics Special Interest Group (BIOSIG) IEEE, Darmstadt, Germany, 10–12 September 2014.
6. Villalba, J.; Lleida, E. Speaker Verification Performance Degradation Against Spoofing and Tampering Attacks. In Proceedings of the FALA Workshop, Vigo, Spain, 10–12 November 2010.
7. Shang, W.; Stevenson, M. Score normalization in playback attack detection. In Proceedings of the IEEE International Conference on Acoustics, Speech & Signal Processing, Dallas, TX, USA, 14–19 March 2010.
8. Zhang, L.; Cao, J.; Xu, M.; Zheng, F. Prevention of impostors entering speaker recognition systems. *J. Tsinghua Un. Sci. Technol.* **2008**, *48*, 699–703.
9. Li, L.; Chen, Y.; Wang, D.; Zheng, T.F. A Study on Replay Attack and Anti-Spoofing for Automatic Speaker Verification. *arXiv* **2017**, arXiv:1706.02101.
10. Todisco, M.; Delgado, H.; Evans, N. Constant Q Cepstral Coefficients: A Spoofing Countermeasure for Automatic Speaker Verification. *Comput. Speech Lang.* **2017**, *45*, 516–535. [CrossRef]
11. Galina, L.; Sergey, N.; Egor, M.; Alexander, K.; Oleg, K.; Vadim, S. Audio replay attack detection with deep learning frameworks. In Proceedings of the Annual Conference of the International Speech Communication Association, Stockholm, Sweden, 20–24 August 2017.
12. Patil, H.A.; Kamble, M.R.; Patel, T.B.; Soni, M. Novel variable length teager energy separation based instantaneous frequency features for replay detection. In Proceedings of the Annual Conference of the International Speech Communication Association, Stockholm, Sweden, 20–24 August 2017.
13. Raffel, C.; Ellis, D.P.W. Feed-forward networks with attention can solve some long-term memory problems. In Proceedings of the International Conference on Learning Representations (ICLR), San Juan, Puerto Rico, 12–19 November 2016.

14. Chan, W.; Jaitly, N.; Le, Q.V.; Vinyals, O. Listen, attend and spell: A neural network for large vocabulary conversational speech recognition. In Proceedings of the IEEE International Conference on Acoustics Speech & Signal Processing (ICASSP), Shanghai, China, 20–25 March 2016.
15. Chorowski, J.; Bahdanau, D.; Serdyuk, D.; Cho, K.; Bengio, Y. Attention-Based Models for Speech Recognition. *Comput. Sci.* **2015**, *10*, 429–439.
16. Zhang, S.; Chen, Z.; Zhao, Y.; Li, J.; Gong, Y. End-to-end attention based text-dependent speaker verification. In Proceedings of the IEEE Spoken Language Technology Workshop (SLT), San Diego, CA, USA, 13–16 December 2016.
17. Hu, D. An Introductory Survey on Attention Mechanism in NLP Problems. *arXiv* **2018**, arXiv:1811.05544.
18. Bahdanau, D.; Cho, K.; Bengio, Y. Neural machine translation by jointly learning to align and translate. In Proceedings of the International Conference on Learning Representations (ICLR), San Diego, CA, USA, 7–9 May 2015.
19. Cho, K. Introduction to Neural Machine Translation with GPUs (part 3). Available online: https://devblogs.nvidia.com/introduction-neural-machine-translation-gpus-part-3/ (accessed on 26 July 2015).
20. Delgado, H.; Todisco, M.; Sahidullah, M.; Evans, N.; Kinnunen, T.; Lee, K.A.; Yamagishi, J. ASVspoof 2017 version 2.0: Meta-data analysis and baseline enhancements. In Proceedings of the Odyssey-the Speaker & Language Recognition Workshop, Les Sables d'Olonne, France, 26–29 June 2018.
21. Steven, D. Robust Speaker Verification System with Anti-spoofing Detection and DNN Feature Enhancement Modules. Master's Dissertation, Nanyang Technological University, Singapore, 2015.
22. ASVspoof 2017: Automatic Speaker Verification Spoofing and Countermeasures Challenge. Available online: http://www.asvspoof.org/data2017/baseline_CM.zip (accessed on 25 March 2017).
23. Sadjadi, S.O.; Slaney, M.; Heck, L. MSR Identity Toolbox: A MATLAB Toolbox for Speaker Recognition Research. *Speech Lang. Tech. Comm. Newsl.* **2013**, *1*, 1–32.
24. Lai, C.I.; Abad, A.; Richmond, K.; Yamagishi, J.; Dehak, N.; King, S. Attentive filtering networks for audio replay attack detection. *arXiv* **2018**, arXiv:1810.13048v1.
25. Valenti, G.; Delgado, H.; Todisco, M.; Evans, N.; Laurent, P. *An End-to-End Spoofing Countermeasure for Automatic Speaker Verification Using Evolving Recurrent Neural Networks*; Odyssey-the Speaker & Language Recognition Workshop: Les Sables d'Olonne, France, 2018.