On the Use of Deep Recurrent Neural Networks for Detecting Audio Spoofing Attacks

Simone Scardapane, Lucas Stoffl, Florian Röhrbein and Aurelio Uncini

Abstract—**Biometric security systems based on predefined speech sentences are extremely common nowadays, particularly in low-cost applications where the simplicity of the hardware involved is a great advantage. Audio spoofing verification is the problem of detecting whether a speech segment acquired from such a system is genuine, or whether it was synthesized or modified by a computer in order to make it sound like an authorized person. Developing countermeasures for spoofing attacks is clearly essential for having effective biometric and security systems based on audio features, all the more significant due to recent advances in generative machine learning. Nonetheless, the problem is complicated by the possible lack of knowledge on the technique(s) used to put forward the attack, so that anti-spoofing systems should be able to withstand also spoofing attacks that were not considered explicitly in the training stage. In this paper, we analyze the use of deep recurrent networks applied to this task, i.e. networks made by the successive combination of multiple feedforward and recurrent layers. These networks are routinely used in speech recognition and language identification but, to the best of our knowledge, they were never considered for this specific problem. We evaluate several architectures on the dataset released for the ASVspoof 2015 challenge last year. We show that, by working with very standard feature extraction routines and with a minimum amount of fine-tuning, the networks can already reach very promising error rates, comparable to state-of-the-art approaches, paving the way to further investigations on the problem using deep RNN models.**

I. Introduction

The field of 'audio biometrics' is concerned with recognizing the identity of a person based on his/her speech patterns [1], [2]. Differently from other commonly used biometric characteristics, e.g. fingerprints or retina scans, security identification via audio is attractive due to the availability of cheap, unobtrusive sensors that can easily be deployed in a security-aware area with low cost. At the same time, the audio recorded from a sensor is sensitive to the presence of background noise, room reverberation, multiple sources and, most importantly for the purpose of this paper, it is subject to a wide range of possible attacks aiming at simulating one person's speech. While noise problems can be solved, in principle, by using high-definition microphones (or possibly arrays of them [3]), understanding whether a recorded speech segment is real or artificial is still an open research problem, restraining a wider adoption of audio biometrics technologies.

Simone Scardapane and Aurelio Uncini are with the Department of Information Engineering, Electronics and Telecommunications (DIET), "Sapienza" University of Rome, Via Eudossiana 18, 00184 Rome, Italy. Lucas Stoffl and Florian Röhrbein are with the Department of Informatics, Technische Universität München, Boltzmannstraße 3, 85748 Garching bei München, Germany. Emails: {simone.scardapane, aurelio.uncini}@uniroma1.it, {florian.roehrbein, lucas.stoffl}@in.tum.de.

Originally introduced in the networking literature, 'spoofing' is a generic term for denoting attacks where an agent tries to impersonate another agent in order to exploit its security credentials [4], [5]. Specifically, we focus on audio synthesis and voice conversion attacks, wherein one or more algorithmic techniques are used to generate an artificial waveform corresponding to a user having access to a specific location or service. The anti-spoofing problem is related to both speaker identification (SI) tasks, where we wish to infer the identity of a speaker among a fixed set of candidates, and to speaker verification (SV) tasks, where the learning system is tuned to understand whether a given utterance was pronounced or not by a given speaker. SI and SV are somewhat equivalent problems, although the SV formulation is generally preferred for evaluation purposes and for customizing a system to a given level of false positive ratio [1].

Owing to the similarity between anti-spoofing and speaker recognition, the two problems have followed similar technological trends over the last decades. In the case of text-independent approaches, state-of-the-art results are commonly obtained by techniques trained to capture the patterns in the frequency/cepstral content of the audio, extracted from short-term fragments (frames) of the original signals. Until a few years ago, inference was generally performed by a combination of Gaussian mixture models (GMMs) [6], discriminative support vector approaches [7], or latent factor analysis (FA) methods [8]. Following the enormous success of deep learning in speech recognition and acoustic modeling [9], very recently the accuracies for the SV problem have been improved by the combination of i-vector systems and deep (feedforward) neural networks [10]–[12]. In this case, deep neural networks can be used both in a direct fashion, by training them to discriminate among spoofing attacks and real audio, or in an indirect way, where they are employed as generic feature extractors for use in further classification routines. In particular, it is common to use 'bottleneck' features extracted in the last layer [12].

One peculiarity of the anti-spoofing problem is that, although we have knowledge of a given set of possible attacks during the training stage, what truly matters during the deployment of the system is robustness to *unknown* spoofing attacks, even outside those considered before. For this reason, every system that is measuring its performance on the same set of attacks used during training will, in all likelihood, severely over-estimate its accuracy when successively used in the real-world. In order to raise awareness on this point and on anti-spoofing in general, lately several challenges have been organized in order to evaluate the state-of-the-art in the field, and to provide a common testbed for the evaluation of further methods. They include the special session on spoofing at the 2013 Interspeech conference [4] and, more recently, the 2015 'Automatic Speaker Verification Spoofing and Countermeasures Challenge' (ASVspoof) in 2015 [13]. These challenges have confirmed that, despite the variability of techniques, results on out-of-sample attacks still have room for definite improvements in performance. This underlines the importance of evaluating new, and more sophisticated, approaches to counter the anti-spoofing problem.

In this paper, we present some initial results on the use of deep recurrent neural networks (RNNs) when applied to anti-spoofing tasks. Differently from standard neural networks, RNNs are able to process temporal data by the use of intra-layer connections, resulting in an internal state keeping information on the previously processed patterns [14]. Specifically, long short term memory (LSTM) networks, originally introduced by Hochreiter and Schmidhuber [15], have achieved remarkable success in speech recognition, owing to their capability of processing long sequences without incurring into (or, at the very least, by mitigating sufficiently) the vanishing and exploding gradient problems [16], [17]. As shown in [18], there are several ways to extend LSTMs in order to obtain explicitly deep models. Specifically, in this paper we consider a deep architecture composed of multiple feedforward layers, followed by one or more layers of LSTM neurons (see Fig. 2 and its description in Section III for additional details). From an intuitive viewpoint, the former set of layers is used to extract meaningful high-order information from the underlying frame representation, while the latter set is used to process the signal at different time resolutions. Our work is motivated by the large accuracy obtained by this family of models in speech recognition problems [19]–[21], where they are rapidly becoming the state-of-the-art. More recently, Huang *et al.* have shown promising results of similar models for source separation in the monaural case [22], further strengthening their applicability in a wide range of tasks.

In order to keep our conclusions as simple as possible, the preliminary results reported in this paper focus on the direct use of deep RNNs, which are trained in a discriminative fashion to distinguish between real speech and spoofing attacks, without the use of a more sophisticated probabilistic back-end. Despite this simplification, our experimental results on the ASVspoof 2015 dataset show that this kind of networks, even with a minimal amount of fine-tuning and a very small set of features extracted from a frame, can already reach results comparable to the state-of-the-art. This promising result hints at the strong representational power of deep RNNs, indicating them as a highly promising line of research in the immediate future where, as we discuss more at length in the conclusive section, the anti-spoofing problem will probably attract even larger amount of attention from the research community.

The rest of the paper is organized as follows. In Section II we describe the spoofing problem more formally, together with the dataset which is used in our experiments. Section III introduces the main concept of a deep RNN with LSTM layers. Next, we analyze the results of the proposed network in Section IV. Finally, we discuss some currently open problems and future lines of research on the anti-spoofing problem in Section V.

II. Anti-spoofing and the ASVspoof 2015 challenge

As we stated in the introduction, the ASVspoof 2015 challenge had the duplicitous objective of (i) providing a common benchmark to test anti-spoofing architectures and (ii) evaluate the state-of-the-art in term of accuracy. In this section we provide a brief description of the task and of the corresponding dataset, and we refer to [13] and to the challenge website[1] for further information. The dataset is freely available in the Edinburgh DataShare repository.[2]

The focus of the challenge was on two broad classes of spoofing attacks. The first one is composed by voice conversion (VC) attacks, wherein a given speech utterance is modified by signal processing techniques to sound as if it was spoken by a different user, which is denoted as the *target* speaker [27]. The second group is composed of speech synthesis (SS) techniques, which is concerned with the generation of speech segments from a corpus of information on the target speaker [25]. Other, slightly more challenging groups of attacks are not considered in the challenge, including impersonation attacks (where a user tries to mimic the voice of the target speaker); replay attacks [29], consisting in the use of pre-recorded speech samples, either surreptitiously or by concatenating multiple speech frames; and spoofing attacks employing artificial, non-speech like signals [30]. We will come back on this point in the conclusive section.

The dataset for the challenge is built from a previous database known as the 'Spoofing and Anti-Spoofing' (SAS) corpus [31]. The genuine portion of the dataset is recorded in a clean environment from 45 males and 61 females. In order to keep the challenge as focused as possible, there are no significants noise effects from the background. Next, the genuine utterances are modified according to 10 different VC and SS spoofing attacks, a brief description of which is provided in Table I. The algorithms designed for the challenge should have no *a priori* knowledge on the attack under consideration and, more importantly, they should work suitably well even for attacks that were not explicitly considered during the training phase. For this reason, the attacks in Table I are split into two groups, depending on whether they are known or unknown before the evaluation phase.

As is common for these challenges, the dataset is then subdivided into three parts. The **training part** is composed by 3750 genuine and 12625 spoofed utterances, selected from a subset of 10 males and 25 females of the entire dataset. The spoofed utterances are created by the application of one of the five known attacks in Table I which, generally speaking, are also the most simple to implement. This portion of the

[1]http://www.spoofingchallenge.org/

[2]http://datashare.is.ed.ac.uk/handle/10283/853

TABLE I

Description of the 10 spoofing attacks considered in the ASVspoof 2015 challenge. Attacks S1-S5 are known during the training phase, while the more complex attacks S6-S10 are only available during the evaluation stage.

Type	ID	Description	Reference
Known	**S1**	Simple frame selection algorithm	[23]
	S2	Simple voice conversion algorithm (only works with the first MFCC value)	[24]
	S3	Speech synthesis algorithm based on the use of HMM and speaker adaptation techniques	[25]
	S4	Same as S3, using more data for the adaptation process	[13]
	S5	Voice conversion algorithm based on the festvox project[3]	NA
Unknown	**S6**	Voice conversion based on GMM and ML parameter adaptation	[26]
	S7	Similar to S6, using a different feature representation	[13]
	S8	Voice conversion based on tensor decomposition with a Japanese dataset	[27]
	S9	Voice conversion based on a kernel partial least-squares algorithm	[28]
	S10	Speech synthesis using the Mary text-to-speech system[4]	NA

dataset can be used to train the classifier of choice. The **development part** of the dataset is composed by selecting 3497 genuine utterances and 49875 spoofed utterances from a different subset of 15 males and 20 females, according to the same spoofing procedures employed for the training part. This portion can be used for designing and optimizing the overall system, e.g. by fine-tuning the hyperparameters via a grid search procedure. Meta information on the speakers is available for the training and development parts, albeit it was not considered for the design of the deep RNNs in this paper. Finally, 9404 genuine utterances and 184000 spoofed utterances were constructed from the remaining subset of 20 males and 26 females and incorporated in the final **test set**, which is only available at the evaluation stage. As stated earlier, spoofed speech in this set is obtained via the application of one of the five spoofing algorithms considered in the previous two parts, or via one of the additional five unknown spoofing algorithms described in the second part of Table I. A schematic description of the available utterances for each spoofing algorithm in the three subsets of the dataset is provided in Table II.

After training, the algorithms are evaluated by presenting them with the speech signals in the test set in a random order, and they are tasked to provide a confidence score on whether the given utterance is genuine or spoofed. Scores close to 0 indicate high probability of spoofed speech, while large scores denote high probability of genuine speech. We can define a false alarm probability given an alarm threshold θ as follows [13]:

$$P_{\text{fa}}(\theta) = \frac{\#\left\{\text{spoofed trials with score} > \theta\right\}}{\#\left\{\text{total spoofed trials}\right\}} \,. \tag{1}$$

In a similar way, we can define a probability of missing a spoofed utterance as:

$$P_{\text{miss}}(\theta) = \frac{\#\left\{\text{genuine trials with score} \leq \theta\right\}}{\#\left\{\text{total genuine trials}\right\}} \,. \tag{2}$$

The error measure is chosen as the equal error rate (EER), which is defined by choosing a value θ^* for the threshold such that $P_{\text{fa}}(\theta^*) = P_{\text{miss}}(\theta^*) = \text{EER}$. 10 different values of EERs are computed, one for each spoofing algorithm independently, and their average is taken as the final error measure for the system.

In order to evaluate the complexity of the overall task, a baseline system was constructed for the challenge, based on a probabilistic version of linear discriminant analysis [32], obtaining an average EER of 36% for male speakers, and of 39.53% for the female speakers in the evaluation set. The S10 attack, due to the complexity of the underlying software, is particularly challenging, resulting in a baseline EER of 51.17% for male speakers, and of 44.20% for the female speakers in

TABLE II

Details on the available utterances, divided by spoofing algorithm and dataset portion. In the fifth column, VC stands for voice conversion, while SS stands for speech synthesis.

Attack	**Segments available**			**Spoofing type**
	Train	Development	Evaluation	
Genuine	3750	3497	9404	NA
S1	2525	9975	18400	VC
S2	2525	9975	18400	VC
S3	2525	9975	18400	SS
S4	2525	9975	18400	SS
S5	2525	9975	18400	VC
S6	0	0	18400	VC
S7	0	0	18400	VC
S8	0	0	18400	VC
S9	0	0	18400	VC
S10	0	0	18400	SS

[3]http://www.festvox.org/

[4]http://mary.dfki.de/

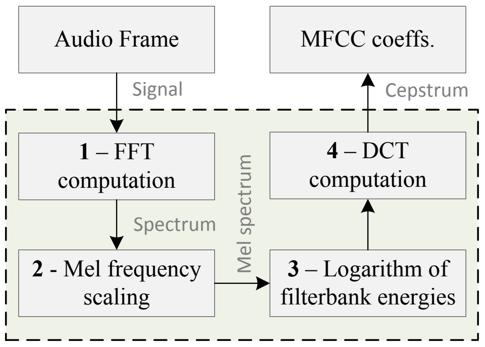

Fig. 1. Main steps involved in the extraction of MFCC coefficients from a generic audio frame, highlighted in a green background.

the evaluation set. Results of the best performing algorithms in the challenge will be discussed later on when comparing them with our proposed architecture.

III. Proposed architecture

This section describes the proposed architecture for the anti-spoofing problem. Specifically, we summarize the baseline feature extraction procedure we employ in Section III-A, while the deep RNN formulation is provided in Section III-B. Details on the training procedure are given in the subsequent section.

A. Feature extraction

Each audio segment is decomposed into frames of 100 ms duration, with no overlap. Using this setup, the longest audio segment results in a total of 1323 frames, while the use of overlapping frame extraction would have exceeded the available hardware constraints (see next section for more details). Classification is performed on two set of features based on the very popular Mel frequency cepstral coefficients (MFCCs), which are a low-level, timbre-based representation of the signal [33]. For completeness, we summarize here the main steps involved in their extraction as shown in Fig. 1. First, a fast Fourier transform (FFT) algorithm is applied to a frame decomposition of the signal to extract the spectral representation of each frame. Denoting by $s[k]$ the kth complex value of the FFT, its power spectrum is computed as:

$$p[k] = \frac{1}{T} \left| s[k] \right|^2 , \tag{3}$$

where T is the length of the frame. Next, the power spectrum coefficients are rescaled according to a rough model of the human auditory system, by convoluting them with a set of 26 triangular filters whose width increases proportionally to the frequency under consideration. After a logarithm is applied to the resulting filterbank set of features, a discrete cosine transform (DCT) is used to extract the final MFCC coefficients. At this point, all but the first 13 coefficients are discarded to obtain the final representation. In our experimental section, we also consider the use of the MFCCs supplemented by the log-filterbank energies extracted at step 3, and the log-filterbank features alone.

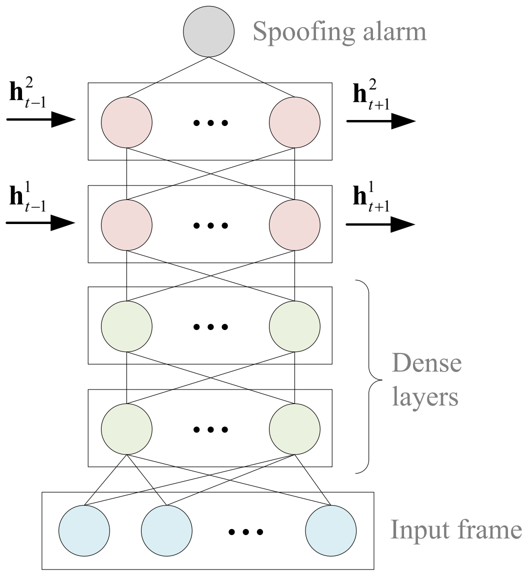

Fig. 2. Schematic visualization of the network's architecture with an input layer (blue), two dense layers (green), and two recurrent layers (red). The recurrent layers also receive information from the previous time step.

B. Network model

The feature vectors extracted from every audio segment are sequentially analyzed by a deep RNN model, an example of which is shown in Fig. 2. Specifically, the network is composed by a stack of N fully connected feedforward models (shown in green in Fig. 2), followed by M recurrent layers composed of LSTM neurons (shown in red in Fig. 2). In order to set the notation, we denote by $\mathbf{a}_t^i$ the input of the ith feedforward layer at time t, with $i = 1, \ldots, N$, where $\mathbf{a}_t^1$ is the cepstral representation extracted as in the last section. The output of a generic feedforward layer is computed as:

$$\mathbf{a}_t^{i+1} = r(\mathbf{W}^i \mathbf{a}_t^i + \mathbf{b}^i) , \tag{4}$$

where $\mathbf{W}^i$ and $\mathbf{b}^i$ are the adaptable parameters of the layer, and we use ReLU nonlinearities defined element-wise as:

$$r(x) = \max\{0, x\} . \tag{5}$$

The vector $\mathbf{a}_t^{N+1}$ is then propagated through M additional recurrent layers. In order to maintain a consistent notation, we let $\mathbf{a}_t^{N+i}$ define the input of the generic ith recurrent LSTM layer, with $i = 1, \ldots, M$, by $\mathbf{h}_{t-1}^i$ its state at the previous time instant, and similarly by $\mathbf{C}_{t-1}^i$ its cell state. A single LSTM layer is defined as follows [15]. First, we compute the block input of the layer as:

$$\mathbf{g}_t^i = \sigma\left(\mathbf{W}_{\text{in}}^i \mathbf{a}_t^{N+i} + \mathbf{U}_{\text{in}}^i \mathbf{h}_{t-1}^i + \mathbf{b}_{\text{in}}^i\right) , \tag{6}$$

where $\mathbf{W}_{\text{in}}^i$, $\mathbf{U}_{\text{in}}^i$ and $\mathbf{b}_{\text{in}}^i$ are adapted in the training process, and $\sigma(\cdot)$ is a sigmoid nonlinearity. In a similar fashion, we can compute the input gate activations, describing how much each input will be propagated at the current step:

$$\widetilde{\mathbf{C}}_t^i = \tanh\left(\mathbf{W}_{\text{c}}^i \mathbf{a}_t^{N+i} + \mathbf{U}_{\text{c}}^i \mathbf{h}_{t-1}^i + \mathbf{b}_{\text{c}}^i\right) , \tag{7}$$

TABLE III
A REPRESENTATIVE SET OF RESULTS OBTAINED BY VARYING THE TOPOLOGY OF THE NETWORK AND THE SET OF INPUT FEATURES. IN THE SECOND COLUMN, THE NUMBERS BETWEEN BRACKETS REPRESENT THE SIZE OF THE CORRESPONDING LAYER. 'DENSE' REFERS TO A PURELY FEEDFORWARD LAYER AS IN (4).

Feature set	**Topology**	**Equal Error Rate (EER) [%]**		
		Known (S1-S5)	Unknown (S6-S10)	Average
MFCC	3x Dense (13, 12, 10) + 3x LSTM (10, 20, 30)	$(2.2, 3.4, 0.0, 0.2, 3.5)$	$(3.9, 2.4, 0.0, 2.8, 10.7)$	2.91
Log-filterbank	3x LSTM (64, 64, 64)	$(15.2, 15.3, 15.0, 15.0, 15.3)$	$(15.3, 15.3, 15.0, 15.4, 37.7)$	17.25
MFCC + log-filterbank	3x LSTM (128, 128, 128)	$(6.5, 9.0, 4.4, 4.2, 10.1)$	$(10.3, 7.5, 2.8, 8.4, 38.1)$	10.14
MFCC + log-filterbank	3x Dense (39, 37, 35) + 3x LSTM (35, 30, 25)	$(0.3, 0.7, 0.4, 0.4, 0.9)$	$(0.9, 0.6, 0.5, 0.7, 96.0)$	10.15

where again $\mathbf{W}_\text{c}^i$, $\mathbf{U}_\text{c}^i$ and $\mathbf{b}_\text{c}^i$ are adaptable sets of parameters. Next, we compute the activation of the *forget gate* $\mathbf{f}_t^i$, which decides how much every element of the internal cell state will be updated. The computation is equivalent to (6) with a different set of adaptable parameters and it is omitted for brevity. The new cell state is then given by:

$$\mathbf{C}_t^i = \mathbf{g}_t^i \odot \widetilde{C}_t^i + \mathbf{f}_t^i \odot \mathbf{C}_{t-1}^i \,, \tag{8}$$

where $\odot$ denotes the element-wise product. Finally, the output gate and the new states are produced as:

$$\mathbf{o}_t^i = \sigma\left(\mathbf{W}_\text{o}^i \mathbf{a}_t^{N+i} + \mathbf{U}_\text{o}^i \mathbf{h}_{t-1}^i + \mathbf{V}_\text{o}^i \mathbf{C}_t^i + \mathbf{b}_\text{o}^i\right) \tag{9}$$

$$\mathbf{h}_t^{N+i} = \mathbf{o}_t^i \odot \tanh\left(\mathbf{C}_t^i\right) \,, \tag{10}$$

where $\mathbf{W}_\text{o}^i$, $\mathbf{U}_\text{o}^i$, $\mathbf{V}_\text{o}^i$ and $\mathbf{b}_\text{o}^i$ are the final adaptable parameters of the layer, and the layer output $\mathbf{a}_t^{N+i+1}$ is given by the new state $\mathbf{h}_t^{N+i}$. Multiple variations over this basic architecture are possible, such as the use of additional 'peephole' connections in the input, forget and output gates, or by considering simplified architectures such as the gated recurrent unit (GRU), that combines input and forget gate into a single update block [34]. We leave the exploration of these models for the anti-spoofing task to a future work.

IV. EXPERIMENTAL RESULTS

A. Setup

The deep RNN model is implemented in Python, using the Theano[5] and Lasagne[6] libraries. Frame decomposition and feature extraction is performed with the python_speech_features module.[7] For training the network, we minimize the mean-squared error on the train set, weighted by an ℓ_2 norm regularization on the parameters of the network. We also experimented with a cross-entropy formulation, although no significant improvements in accuracy were found.

Gradients are computed via a truncated backpropagation through time, where the output on an audio segment is defined by averaging the final 25 outputs of the network for the corresponding sequence, in order to increase stability. Updates are performed on mini-batches of 500 utterances with the Adam optimization algorithm [35], where all parameters are kept to their default values. Optimization is performed for a maximum of 150 epochs, while the development set is used to select an optimal regularization term in an exponential range. We randomly dropout neurons in the feedforward layers with probability 10% during training [36]. Naïve dropout on the LSTM layers is known to worsen performance, although we plan to investigate more efficient ways of applying it in future works, e.g. following the procedure in [37]. Future work can also consider a more elaborate fine-tuning procedure, e.g. using Bayesian optimization [38], [39]. Experiments are performed on an Intel® Xeon E5-2620 @ 2.10 GHz, with 8 GB of RAM and a CUDA back-end employing an Nvidia Tesla K20c. All results are averaged over 15 different initializations of the weights.

B. Results

We experiment with different combinations of feedforward (from here on, denoted as 'dense' layers), and LSTM layers, in combination with MFCC features, log-filterbank features, or a concatenation of both, trying to cover all the spectrum of possible variations. A representative set of results is reported in Table III. As expected, in all cases the most challenging attack is the unknown S10 attack.

3 LSTM layers trained on the log-filterbank features only (second row of Table III) are performing relatively poorly, while including MFCC features in this setting (third row of Table III) slightly increases the performance. The best result globally, when also considering S10, is however reached by the combination of 3 dense layers and 3 LSTM layers trained on the MFCC features only (first row of Table III), which has a definite reduction in EER for attacks S1-S9 and a similarly drastic reduction of EER for attack S10. By including also log-filterbank features (fourth row of Table III) we reduces the EER on attacks S1-S9, but the S10 attack is almost never recognized. This is a case of 'in-sample' overfitting, wherein the trained network is not able to generalize efficiently to

[5]http://deeplearning.net/software/theano/
[6]https://github.com/Lasagne/Lasagne
[7]https://pypi.python.org/pypi/python_speech_features

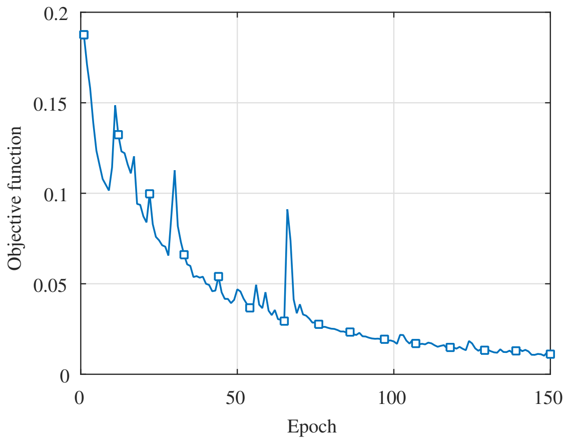

Fig. 3. Objective function evolution on a representative run.

attacks that are too different from those in the train set. This actually suggests the possibility that using an ensemble of different networks, like is routinely done in speech recognition [40], can be a viable option to combine the advantages of deep RNNs on all types of attacks. We leave this open as an additional line of investigation. A representative plot of the cost function evolution over the different epochs is provided in Fig. 3 for a randomly selected trial. We see that a satisfactory level of accuracy is already reached around epoch 75.

An interesting question is how well the deep RNN architectures are faring with respect to the top performing algorithms of the challenge. For comparison, we report the top 5 primary submissions, including the overall winner, in Table IV, where the systems are identified by the same letter as in the challenge's website. We underline again that these results are obtained with extremely more sophisticated architectures than those considered here employing, among other, complex probabilistic back-ends and combinations of multiple set of orthogonal features. Nonetheless, we see that our best performing deep RNN, which is trained directly on the MFCC acoustic features, is actually performing very close to the submissions C, D and E. In fact, its average EER on unknown attacks puts it on a par with the second best submission, and much higher than all the rest of the models. It is interesting to observe from Table IV that, again, very good results on unknown attacks (like for submission A) are obtained by partially compromising the EER on known attacks.

V. Discussion and conclusive remarks

In this paper we have shown that deep RNN architectures can reach state-of-the-art results in the anti-spoofing task, even with a small number of layers and a minimal amount of fine-tuning. This paves the way to the possibility of further increasing their accuracy over the current known results in a variety of ways, e.g. by considering additional layers, increasing the amount of input features, training an ensemble of models, and inserting a probabilistic back-end to elaborate the outputs of the network. Even more interestingly, we can think of training the model on the raw audio data to see what features the network can learn in order to discriminate among spoofing and non-spoofing attacks. Parallel to this, we envision their use on more complex spoofing attacks, such as those discussed more at length in Section II, or a combination of several basic attacks. Deep RNNs can also be integrated in existing voice recognition, language identification, and speaker classification systems, resulting in extremely smart multitask models all exploiting a common infrastructure.

More in general, we conjecture that the development of novel approaches to the anti-spoofing problem will be crucial in the coming years to ensure the efficiency of biometric systems, due to the recent rise of interest in generative machine learning models. As an example, Oord *et al.* [46] have recently shown that, with an appropriate availability of hardware resources, it is feasible to directly work with the entire raw waveform of an audio signal at once, resulting in extremely convincing speech over a variety of languages and speakers. We imagine that this kind of result in speech synthesis will play a significant role in the design of more sophisticated spoofing attacks for which the current state-of-the-art is not sufficient. Thus, exploring new avenues of research exploiting ever deeper neural models will be a central point in future anti-spoofing research.

References

[1] P. S. Aleksic and A. K. Katsaggelos, "Audio-visual biometrics," *Proceedings of the IEEE*, vol. 94, no. 11, pp. 2025–2044, 2006.

[2] N. Evans, T. Kinnunen, J. Yamagishi, Z. Wu, F. Alegre, and P. De Leon, "Speaker recognition anti-spoofing," in *Handbook of Biometric Anti-Spoofing*. Springer, 2014, pp. 125–146.

[3] D. Pavlidi, A. Griffin, M. Puigt, and A. Mouchtaris, "Real-time multiple sound source localization and counting using a circular microphone array," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 21, no. 10, pp. 2193–2206, 2013.

[4] N. W. Evans, T. Kinnunen, and J. Yamagishi, "Spoofing and countermeasures for automatic speaker verification." in *Interspeech 2013*, 2013, pp. 925–929.

[5] Z. Wu, N. Evans, T. Kinnunen, J. Yamagishi, F. Alegre, and H. Li, "Spoofing and countermeasures for speaker verification: a survey," *Speech Communication*, vol. 66, pp. 130–153, 2015.

[6] D. A. Reynolds, T. F. Quatieri, and R. B. Dunn, "Speaker verification using adapted gaussian mixture models," *Digital signal processing*, vol. 10, no. 1, pp. 19–41, 2000.

[7] W. M. Campbell, D. E. Sturim, and D. A. Reynolds, "Support vector machines using gmm supervectors for speaker verification," *IEEE Signal Processing Letters*, vol. 13, no. 5, pp. 308–311, 2006.

[8] N. Dehak, P. J. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Front-end factor analysis for speaker verification," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 4, pp. 788–798, 2011.

[9] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-R. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath *et al.*, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[10] Y. Lei, N. Scheffer, L. Ferrer, and M. McLaren, "A novel scheme for speaker recognition using a phonetically-aware deep neural network," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2014, pp. 1695–1699.

[11] P. Karanasou, Y. Wang, M. J. Gales, and P. C. Woodland, "Adaptation of deep neural network acoustic models using factorised i-vectors." in *Interspeech 2014*, 2014, pp. 2180–2184.

[12] F. Richardson, D. Reynolds, and N. Dehak, "Deep neural network approaches to speaker and language recognition," *IEEE Signal Processing Letters*, vol. 22, no. 10, pp. 1671–1675, 2015.

TABLE IV
RESULTS OF THE BEST 5 SUBMISSIONS AT THE ASVSPOOF 2015 CHALLENGE, TOGETHER WITH A SHORT DESCRIPTION, REPORTED FOR COMPARISON WITH RESPECT TO THE PROPOSED ARCHITECTURE. BEST RESULTS FOR EACH COLUMN ARE HIGHLIGHTED IN BOLD.

System	**Equal Error Rate (EER) [%]**			**Description**	**Reference**
	Known (S1-S5)	Unknown (S6-S10)	Average		
A	0.408	**2.013**	**1.211**	GMM classification based on cochlear filters	[41]
B	0.008	3.922	1.965	Total Variability Joint FA on multiple feature sets	[42]
C	0.058	4.998	2.528	Mahalanobis distance of ‘deep’ extracted features	[43]
D	**0.003**	5.231	2.617	Shallow NN trained on high-dimensional features	[44]
E	0.041	5.347	2.694	Linear fusion of several GMM-based subsystems	[45]
Proposed	0.540	3.960	2.910	Deep RNN trained on MFCC features alone (first row in Table III)	NA

[13] Z. Wu, T. Kinnunen, N. Evans, J. Yamagishi, C. Hanilçi, M. Sahidullah, and A. Sizov, “Asvspoof 2015: the first automatic speaker verification spoofing and countermeasures challenge,” in *Interspeech 2015*, 2015, pp. 2037–2041.

[14] R. Pascanu, T. Mikolov, and Y. Bengio, “On the difficulty of training recurrent neural networks.” *Proc. of the 2013 Internal Conference on Machine Learning (ICML)*, vol. 28, pp. 1310–1318, 2013.

[15] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[16] F. Eyben, F. Weninger, S. Squartini, and B. Schuller, “Real-life voice activity detection with lstm recurrent neural networks and an application to hollywood movies,” in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2013, pp. 483–487.

[17] M. Sundermeyer, H. Ney, and R. Schlüter, “From feedforward to recurrent lstm neural networks for language modeling,” *IEEE/ACM Transactions on Audio, Speech and Language Processing*, vol. 23, no. 3, pp. 517–529, 2015.

[18] R. Pascanu, C. Gulcehre, K. Cho, and Y. Bengio, “How to construct deep recurrent neural networks,” *arXiv preprint arXiv:1312.6026*, 2013.

[19] A. Graves, N. Jaitly, and A.-R. Mohamed, “Hybrid speech recognition with deep bidirectional lstm,” in *2013 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*. IEEE, 2013, pp. 273–278.

[20] A. Graves, A.-R. Mohamed, and G. Hinton, “Speech recognition with deep recurrent neural networks,” in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2013, pp. 6645–6649.

[21] X. Li and X. Wu, “Constructing long short-term memory based deep recurrent neural networks for large vocabulary speech recognition,” in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2015, pp. 4520–4524.

[22] P.-S. Huang, M. Kim, M. Hasegawa-Johnson, and P. Smaragdis, “Joint optimization of masks and deep recurrent neural networks for monaural source separation,” *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 23, no. 12, pp. 2136–2147, 2015.

[23] T. Dutoit, A. Holzapfel, M. Jottrand, A. Moinet, J. Prez, and Y. Stylianou, “Towards a voice conversion system based on frame selection,” in *2007 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, vol. 4. IEEE, 2007, pp. IV–513–IV–519.

[24] T. Fukada, K. Tokuda, T. Kobayashi, and S. Imai, “An adaptive algorithm for mel-cepstral analysis of speech,” in *1992 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, vol. 1. IEEE, 1992, pp. 137–140.

[25] J. Yamagishi, T. Kobayashi, Y. Nakano, K. Ogata, and J. Isogai, “Analysis of speaker adaptation algorithms for hmm-based speech synthesis and a constrained smaplr adaptation algorithm,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 17, no. 1, pp. 66–83, 2009.

[26] T. Toda, A. W. Black, and K. Tokuda, “Voice conversion based on maximum-likelihood estimation of spectral parameter trajectory,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 15, no. 8, pp. 2222–2235, 2007.

[27] D. Saito, K. Yamamoto, N. Minematsu, and K. Hirose, “One-to-many voice conversion based on tensor representation of speaker space.” in *Interspeech 2011*, 2011, pp. 653–656.

[28] E. Helander, H. Silén, T. Virtanen, and M. Gabbouj, “Voice conversion using dynamic kernel partial least squares regression,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 3, pp. 806–817, 2012.

[29] J.-F. Bonastre, D. Matrouf, and C. Fredouille, “Artificial impostor voice transformation effects on false acceptance rates.” in *Interspeech 2007*, 2007, pp. 2053–2056.

[30] F. Alegre, R. Vipperla, and N. Evans, “Spoofing countermeasures for the protection of automatic speaker recognition systems against attacks with artificial signals,” in *Interspeech 2012*, 2012.

[31] Z. Wu, A. Khodabakhsh, C. Demiroglu, J. Yamagishi, D. Saito, T. Toda, and S. King, “Sas: A speaker verification spoofing database containing diverse attacks,” in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2015, pp. 4440–4444.

[32] S. Prince, P. Li, Y. Fu, U. Mohammed, and J. Elder, “Probabilistic models for inference about identity,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 1, pp. 144–157, 2012.

[33] Z. Fu, G. Lu, K. M. Ting, and D. Zhang, “A survey of audio-based music classification and annotation,” *IEEE Transactions on Multimedia*, vol. 13, no. 2, pp. 303–319, 2011.

[34] K. Greff, R. K. Srivastava, J. Koutník, B. R. Steunebrink, and J. Schmidhuber, “Lstm: A search space odyssey,” *arXiv preprint arXiv:1503.04069*, 2015.

[35] D. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014.

[36] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: a simple way to prevent neural networks from overfitting.” *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[37] W. Zaremba, I. Sutskever, and O. Vinyals, “Recurrent neural network regularization,” *arXiv preprint arXiv:1409.2329*, 2014.

[38] J. Bergstra and Y. Bengio, “Random search for hyper-parameter optimization,” *Journal of Machine Learning Research*, vol. 13, no. Feb, pp. 281–305, 2012.

[39] J. Snoek, H. Larochelle, and R. P. Adams, “Practical bayesian optimization of machine learning algorithms,” in *Advances in neural information processing systems*, 2012, pp. 2951–2959.

[40] L. Deng and J. C. Platt, “Ensemble deep learning for speech recognition.” in *Interspeech 2014*, 2014, pp. 1915–1919.

[41] T. B. Patel and H. A. Patil, “Combining evidences from mel cepstral, cochlear filter cepstral and instantaneous frequency features for detection of natural vs. spoofed speech,” in *Interspeech 2015*, 2015, pp. 2062–2066.

[42] S. Novoselov, A. Kozlov, G. Lavrentyeva, K. Simonchik, and V. Shchemelinin, “Stc anti-spoofing systems for the asvspoof 2015 challenge,” in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 5475–5479.

[43] N. Chen, Y. Qian, H. Dinkel, B. Chen, and K. Yu, “Robust deep feature for spoofing detection-the sjtu system for asvspoof 2015 challenge,” in *Interspeech 2015*, 2015, pp. 2097–2101.

[44] X. Xiao, X. Tian, S. Du, H. Xu, E. S. Chng, and H. Li, “Spoofing speech detection using high dimensional magnitude and phase features: The ntu approach for asvspoof 2015 challenge,” in *Interspeech 2015*, 2015, pp. 2052–2056.

[45] M. J. Alam, P. Kenny, G. Bhattacharya, and T. Stafylakis, “Development of crim system for the automatic speaker verification spoofing and countermeasures challenge 2015,” in *Interspeech 2015*, 2015, pp. 2072–2076.

[46] A. v. d. Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, “Wavenet: A generative model for raw audio,” *arXiv preprint arXiv:1609.03499*, 2016.