INTERSPEECH 2017
August 20–24, 2017, Stockholm, Sweden

Audio replay attack detection with deep learning frameworks

Galina Lavrentyeva[1], *Sergey Novoselov*[1], *Egor Malykh*[1], *Alexander Kozlov*[2], *Oleg Kudashev*[1,2], *Vadim Shchemelinin*[1]

[1]ITMO University, St.Petersburg, Russia
[2]STC-innovations Ltd., St.Petersburg, Russia

{lavrentyeva, novoselov, malykh, kozlov-a, kudashev, shchemelinin}@speechpro.com

Abstract

Nowadays spoofing detection is one of the priority research areas in the field of automatic speaker verification. The success of Automatic Speaker Verification Spoofing and Countermeasures (ASVspoof) Challenge 2015 confirmed the impressive perspective in detection of unforeseen spoofing trials based on speech synthesis and voice conversion techniques. However, there is a small number of researches addressed to replay spoofing attacks which are more likely to be used by non-professional impersonators. This paper describes the Speech Technology Center (STC) anti-spoofing system submitted for ASVspoof 2017 which is focused on replay attacks detection. Here we investigate the efficiency of a deep learning approach for solution of the mentioned-above task. Experimental results obtained on the Challenge corpora demonstrate that the selected approach outperforms current state-of-the-art baseline systems in terms of spoofing detection quality. Our primary system produced an EER of 6.73% on the evaluation part of the corpora which is 72% relative improvement over the ASVspoof 2017 baseline system.

Index Terms: spoofing, anti-spoofing, speaker recognition, replay attack detection, CNN, RNN, ASVspoof

1. Introduction

Biometrics technology has advanced tremendously over the last decade. It is becoming ever more widely used in our daily lives. Voice biometrics by its rights remains one of the priority research directions in this field. Automatic speaker verification (ASV) systems are actively marketed due to their reliability, convenience, low-cost and provided security. ASV systems are widely used in call-centers, interactive voice response (IVR) systems and mobile applications. Their high performance allows to use them for protection of more valuable data, such as confidential account information or transaction confirmation. With the growing importance of secured data, the need in the risks estimation of ASV systems spoofing increases [1].

According to [2] attacks at the microphone and transmission levels of the ASV system generally pose the greatest threat. There are four types of spoofing attacks: impersonation, replay audio (RA), speech synthesis (SS) and voice conversion (VC) [3]. Detection of impersonation can be easily solved by ASV system itself [4]. Automatic Speaker Verification Spoofing (ASVspoof) 2015 Challenge [5] showed impressive results in detection of VC and SS. Compared to these spoofing types replay attacks do not need additional knowledge in audio signal processing and are more likely to be used by non-professional impersonators. However, this problem is usually considered under the restricted conditions. In this regard, the task of developing a replay attack detector that will be able to work in a wide range of conditions, is important.

ASVspoof Challenge 2017 [6] was focused on a ”standalone” replay audio detection task for a text-dependent speaker verification system under ”unseen” conditions. This paper describes the Speech Technology Center (STC) anti-spoofing systems submitted for ASVspoof 2017.

The main aim of this research is the investigation of efficiency of the promising convolutional neural network (CNN) approach for solving the RA detection problem. The success of CNN in classification and recognition tasks, such as video classification [7], image classification [8, 9], face recognition [10] was a powerful motivation to apply such approaches for ASV anti-spoofing tasks.

In [11] authors investigated deep learning frameworks applied to VC and SS spoofing detection in ASVspoof 2015 corpora. Deep neural network (DNN), CNN and recurrent neural network (RNN) architectures were shown to be highly effective for this task. Authors also proposed a CNN+RNN architecture and demonstrated its state-of-the-art performance.

Authors of [12] proposed to use temporal CNN architecture for VC and SS spoofing speech detection and also achieved noticeable results on ASVspoof 2015 corpora.

CNN is often used as a robust feature extractor from the unified shape data, for instance images. This approach can be extended to a variety of audio signal classification tasks by representing the input signal in a time-frequency domain. However, special attention should be paid to the fact that CNN input data should have a unified form. In this case it is necessary to require a unified time-frequency (T-F) shape for each utterance or to apply the windowing procedure to time-frequency data with fixed window size.

A number [13, 14, 15] of well-known architectures showed good results in image classification tasks. In this work we utilized the reduced version of Light CNN architecture [16] based on the usage of the Max-Feature-Map activation (MFM) which is based on Max-Out activation function [17]. Neural network with MFM is capable to choose features which are essential for task solving. According to our hypothesis, such type of networks can be successfully implemented for a audio classification task and, in particular, for anti-spoofing.

2. Baseline systems

2.1. ASVspoof baseline system

The baseline system used in this research is the reference implementation of the state-of-the-art RA detector proposed by the

http://dx.doi.org/10.21437/Interspeech.2017-360

organizers of the ASVspoof 2017. This system is based on the constant Q transform technique which is widely used for music signal processing and for detecting spoofing attacks based on SS and VC [18].

In the Front-End of this system constant Q transform cepstral coefficients (CQCC) are estimated according to the scheme in Figure 1. The Back-End uses a standard 2-class Gaussian Mixture Model (GMM) classifier for genuine and spoofed speech. For each utterance the log-likelihood score is obtained from both models and the final system score is computed as the log-likelihood ratio.

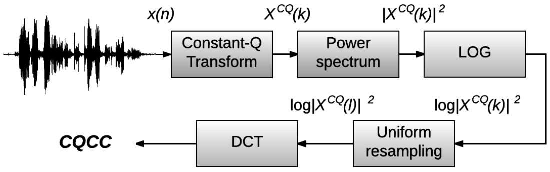

Figure 1: *CQCC features extraction*

As the alternative approach we consider the baseline system with additional steps of mean and variance normalization (MVN) of the log power spectrum and cepstrum.

2.2. I-vector based system

Inspired by the success of our i-vectors based SS and VC spoofing detection system proposed for ASVspoof 2015 [19], we constructed a RA detection system based on the i-vector approach [20].

We experimented with different acoustic features used in the ASVspoof 2015[19]. This system uses Linear Prediction Cepstral Coefficients (LPCC) which provides the best results for RA detection according to our observations. I-vectors are extracted for the whole utterances and are used as an input to the SVM classifier.

Implementation characteristics of both baseline systems are described in details in section 4.

3. Deep learning frameworks

In this section we present deep learning approaches for RA spoofing detection used for high-level features extraction and as End-To-End solution.

3.1. Front-End

We used the normalized log power magnitude spectrum obtained via constant Q transform (CQT) [18] and via Fast Fourier Transform (FFT) as CNN input acoustic features, demonstrated in Figure 2.

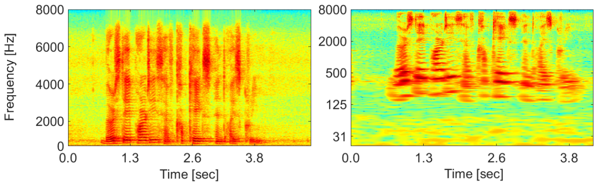

Figure 2: *Log power magnitude spectrum for FFT (left) and CQT (right) for RA with an utterance of phrase "Birthday parties have cupcakes and ice cream"*

We considered two techniques for obtaining a unified time-frequency (T-F) shape of features. One of them truncates the spectrum along the time axis with a fixed size. During this procedure short files are extended by repeating their contents if necessary to match the required length. The other technique uses a sliding window approach with a fixed window size.

3.2. Convolutional neural network

We proposed the spoofing detection method based on CNN with Max-Feature-Map activation. MFM fuction is defined as

$$y_{ij}^k = max(x_{ij}^k, x_{ij}^{k+\frac{N}{2}}),$$
$$\forall i = \overline{1,H}, j = \overline{1,W}, k = \overline{1,N/2}$$

where x is the input tensor of size $H \times W \times N$, y is the output tensor of size $H \times W \times \frac{N}{2}$. Here i, j indicates the frequency and time domains and k is the channel index. Figure 3 illustrates MFM for a convolutional layer. MFM usage allowed us to reduce CNN architecture. That's why such CNN architecture is called Light CNN (LCNN) [16].

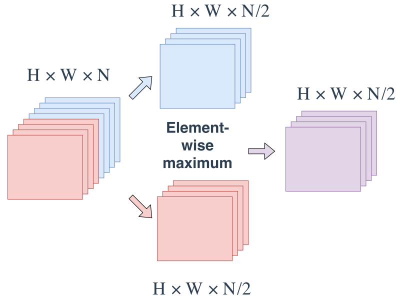

Figure 3: *MFM for convolutional layer*

In contrast to commonly used Rectified Linear Unit function that suppresses a neuron by a threshold (or bias), MFM suppresses a neuron by a competitive relationship. By doing so the MFM plays a role of feature selector.

We used the reduced version of the CNN proposed in [16] with a smaller number of filters in each layer (see Table 1). CNN consists of 5 convolution layers, 4 Network in Network (NIN) layers [21], 10 Max-Feature-Map layers, 4 max-pooling layers and 2 fully connected layers.

Each convolution layer is a combination of two independent convolutional parts calculated of layer's input. Max-Feature-Map activation function is used then to calculate element-wise maximum of these parts. Max-Pooling layers with kernel of size 2 × 2 and stride of size 2 × 2 are used for both time and frequency dimensions reduction. The fully-connected (FC) FC6 layer contains a low-dimensional high-level audio representation. The FC7 layer with softmax activation function is used then to discriminate between genuine and spoofing classes during the training process.

Described CNN was used to obtain high-level audio features. Simple GMM based classifier (see Section 2) can be used to discriminate genuine and spoof classes in this low-dimensional space at evaluation.

3.3. Stacking convolutional and recurrent neural networks

Following the work [11], we use the combined CNN + RNN architecture. The main idea of this integration is that CNN is

Table 1: *CNN architecture*

Type	Filter / Stride	Output	#Params
Conv1	$5 \times 5 / 1 \times 1$	$864 \times 400 \times 32$	832
MFM1	–	$864 \times 400 \times 16$	–
MaxPool1	$2 \times 2 / 2 \times 2$	$432 \times 200 \times 16$	–
Conv2a	$1 \times 1 / 1 \times 1$	$432 \times 200 \times 32$	544
MFM2a	–	$432 \times 200 \times 16$	–
Conv2b	$3 \times 3 / 1 \times 1$	$432 \times 200 \times 48$	7.0K
MFM2b	–	$432 \times 200 \times 24$	–
MaxPool2	$2 \times 2 / 2 \times 2$	$216 \times 100 \times 24$	–
Conv3a	$1 \times 1 / 1 \times 1$	$216 \times 100 \times 48$	1.2K
MFM3a	–	$216 \times 100 \times 32$	–
Conv3b	$3 \times 3 / 1 \times 1$	$216 \times 100 \times 64$	13.9K
MFM3b	–	$216 \times 100 \times 32$	–
MaxPool3	$2 \times 2 / 2 \times 2$	$108 \times 50 \times 32$	–
Conv4a	$1 \times 1 / 1 \times 1$	$108 \times 50 \times 64$	2.1K
MFM4a	–	$108 \times 50 \times 32$	–
Conv4b	$3 \times 3 / 1 \times 1$	$108 \times 50 \times 32$	9.2K
MFM4b	–	$108 \times 50 \times 16$	–
MaxPool4	$2 \times 2 / 2 \times 2$	$54 \times 25 \times 16$	–
Conv5a	$1 \times 1 / 1 \times 1$	$54 \times 25 \times 32$	544
MFM5a	–	$54 \times 25 \times 16$	–
Conv5b	$3 \times 3 / 1 \times 1$	$54 \times 25 \times 32$	4.6K
MFM5b	–	$54 \times 25 \times 16$	–
MaxPool5	$2 \times 2 / 2 \times 2$	$27 \times 12 \times 16$	–
FC6	–	32×2	332K
MFM6	–	32	–
FC7	–	2	64
Total	–	–	371K

used as a feature extractor and RNN models the long-term dependencies. Both CNN and RNN are optimized jointly through the backpropagation, being the End-to-End solution. The overall architecture is shown in Table 2.

Unlike LCNN in 3.2 max-pooling operations are performed using the stride of size 2 along the frequency axis to compress frequency information and the stride of size 1 along the time axis to save time dimensionality. The output of CNN contains 8 channels of size 32×400 each.

The RNN part consists of two gated recurrent units [22] forming the bidirectional gated recurrent unit (BGRU). The first GRU is responsible for the forward pass and processes data from the first input vector to the last one. The second GRU processes data from the last input vector to the first one making backward pass. The last output vectors of both forward and backward passes are taken further to obtain two 16-dimensional vectors. Such BGRU unit is applied to each channel of CNN's output resulting in $16 \times 2 \times 8$ tensor. Weights are shared between every channel's unit to prevent overfitting. Schematic illustration of BGRU is shown in Figure 4.

The flattened output of RNN is used as an input to the fully-connected layers with MFM activations resulting in probability of the utterance being spoofed.

Table 2: *CNN+RNN architecture*

Type	Filter / Stride	Output	#Params
Conv1	$5 \times 5 / 1 \times 1$	$256 \times 400 \times 16$	416
MFM1	–	$256 \times 400 \times 8$	–
MaxPool1	$2 \times 2 / 2 \times 1$	$128 \times 400 \times 8$	–
Conv2a	$1 \times 1 / 1 \times 1$	$128 \times 400 \times 16$	144
MFM2a	–	$128 \times 400 \times 8$	–
Conv2b	$3 \times 3 / 1 \times 1$	$128 \times 400 \times 32$	2.3K
MFM2b	–	$128 \times 400 \times 16$	–
MaxPool2	$2 \times 2 / 2 \times 1$	$64 \times 400 \times 16$	–
Conv3a	$1 \times 1 / 1 \times 1$	$64 \times 400 \times 32$	544
MFM3a	–	$64 \times 400 \times 16$	–
Conv3b	$3 \times 3 / 1 \times 1$	$64 \times 400 \times 16$	2.3K
MFM3b	–	$64 \times 400 \times 8$	–
MaxPool3	$2 \times 2 / 2 \times 1$	$32 \times 400 \times 8$	–
BGRU	–	$16 \times 2 \times 8$	40.3K
FC4	–	512×2	263K
MFM4	–	512	–
FC5	–	256×2	263K
MFM5	–	256	–
FC6	–	1	257
Total	–	–	572K

4. Experimental setup

4.1. Datasets

All experiments in this work were conducted on ASVspoof 2017 datasets. The detailed description of these datasets can be found in [6]. To train all systems considered in this paper we used only the train part. The dev part was used for performance validation and weights adjustment for system fusion. The eval part includes new speakers, environments, replay-recording device combinations and novel attacks that differ substantially from those in the train and dev parts. Therefore, comparison of the proposed systems on the eval part is the most representative.

4.2. Details of systems implementation

Baseline. The ASVspoof baseline system used 29 CQCC, 0-th order cepstral coefficient with first and second derivatives. For the Back-End part two 512-component GMM models were trained for genuine and spoof speech respectively using an expectation-maximisation (EM) algorithm with random initialisation.

SVM i-vector. In the i-vector based baseline system we used 78 LPCC coefficients obtained by using the Hanning window function with a 0.128 sec window size and a 0.016 sec step for FFT power spectrum estimation [23]. In this system the 128-component diagonal covariance UBM was used and the i-vector size was 200. These i-vectors were centered and length-normalized. For the Back-End we used the SVM classifier with a linear kernel [24].

LCNN. We compared 3 LCNN systems. In the $LCNN_{FFT}$ system we used truncated normalized FFT spectrograms of size $864 \times 400 \times 1$ as the input of the first convo-

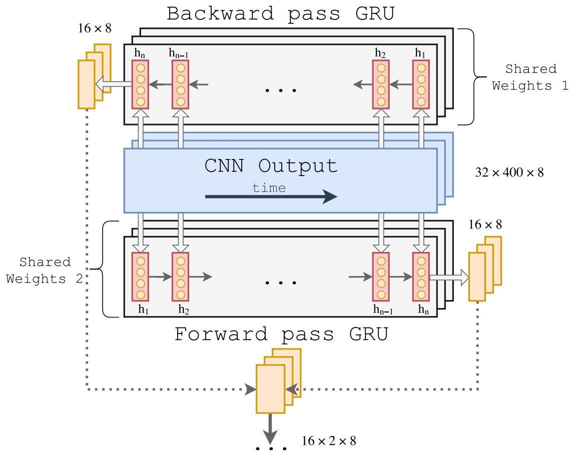

Figure 4: *Bidirectional GRU*

Table 3: *Results on the ASVspoof database*

	EER (%)	
Individual system	**Dev dataset**	**Eval dataset**
$Baseline$	10.35	30.60
$Baseline_{MVN}$	9.85	17.31
SVM_{i-vect}	9.80	12.54
$LCNN_{FFT}$	**4.53**	**7.37**
$LCNN_{FFT}^{SW}$	5.25	11.81
$LCNN_{CQT}$	4.80	16.54
$CNN_{FFT} + RNN$	7.51	10.69
Fusion system		
$LCNN_{FFT}, SVM_{i-vect},$ $CNN_{FFT} + RNN$	**3.95**	**6.73**

lution layer of LCNN.

The architecture of the used LCNN is described in Table 1. Note that the matrix of FC6 layer is very large. In order to prevent over-fitting the dropout with 0.7 ratio was used. Xavier initialization was used for convolutional layers [25]. ADAM optimizer [26] with momentum of 0.9 and learning rate of 10^{-4} was used for training process. Since genuine and replay spoofing classes are well-separable in the obtained high-level feature space, it is enough to use the simple Gaussian models for each class.

Additionally, we explored CQT based features instead of FFT in the $LCNN_{CQT}$ system.

In the $LCNN_{FFT}^{SW}$ system we used a sliding window of $864 \times 200 \times 1$ size and 90% overlapping along time axis to obtain a unified shaped cuts of FFT spectrum. In this scenario we extracted high level features for each window independently. All high level features corresponding to a test utterance were used to estimate the GMM likelihood ratio score.

CNN+RNN. This system used truncated features extracted from the log magnitude power FFT spectrum. Due to the limited computational resources we were forced to reduce the dimension of the input data for the CNN+RNN system. Features were extracted using the Blackman window function with the window size of 256 and the step size of 64. Obtained input tensors for the CNN+RNN system had the size of $256 \times 400 \times 1$.

5. Results and discussion

Table 3 demonstrates resulting EER estimates (%) for all mentioned systems. The results on the dev and eval sets vary greatly due to different conditions. Usage of MVN for the Baseline system features lead to improvement in spoofing detection quality on both dev and eval sets. The i-vector base system showed comparable to baseline results on the dev dataset and demonstrated the stability in detection of spoofing with ”unseen” conditions from the eval part.

The best result for both dev and eval was demonstrated by a single LCNN system used FFT truncated features. It is interesting that the similar system with CQT-based features showed poor stability on the eval set, which may be due to the poor robustness of the CQT features. This is also approved by the results of the baseline system.

The sliding window technique demonstrated worse results compared to the truncated approach on the eval set. A possible reason for this is that using spectrograms of the whole utterances (in most cases) as CNN input leads to more a accurate text-dependent deep model. Our version of using RNN combined with CNN also performed worse than a single LCNN. We explain performance degradation by the reduced frequency resolution in the spectrum estimation.

Our primary system proposed on the ASVspoof 2017 Challenge was presented as a fusion of $LCNN_{FFT}$, SVM_{i-vect} and $CNN_{FFT} + RNN$ systems at the score level [27]. This system demonstrated 3.95% EER and 6.73% EER on the dev and eval set respectively.

In the process of CNN training, we were interested in frequency region that is more informative for genuine and replay speech separating. It turned out that in case of the lower spectrum half (from 0 to 4000 Hz) the achieved accuracy of spoofing detection was 68% on the dev dataset. And in case of the upper frequency band (from 4000 to 8000 Hz), we achieved 85% accuracy on the same validation set.

We suggested that phrase-dependent systems can perform a higher accuracy compared to the system trained for all phrases. We examined the $LCNN_{FFT}$ and i-vector based systems trained for several different phrases independently using the data from the training part. However, our experiments showed reduction in spoofing detection in comparison to the common systems trained on the whole training part of the challenge data. This can be explained by fast overfitting due to the insufficient size of the training data in a phrase-dependent case.

6. Conclusions

In this paper we explored the applicability of the deep learning approach for solution the problem of replay attack spoofing detection. We investigated single CNN and combined with RNN approaches. Our experiments conducted on the ASVspoof 2017 dataset confirmed high efficiency of deep learning frameworks for spoofing detection ”in the wild”. EER of the best individual CNN system was 7.34%. Our primary system based on systems score fusion provided 6.73% EER on the eval set.

7. Acknowledgements

This work was financially supported by the Ministry of Education and Science of the Russian Federation, Contract 14.578.21.0189 (ID RFMEFI57816X0189).

8. References

[1] V. Shchemelinin and K. Simonchik, “Study of voice verification system tolerance to spoofing attacks using a text-to-speech system,” *Instrument Engineering*, vol. 57, no. 2, pp. 84–88, 2014, in Russian.

[2] M. Faundez-Zanuy, “On the vulnerability of biometric security systems,” *IEEE Aerospace and Electronic Systems Magazine*, vol. 19, no. 6, pp. 3–8, June 2004.

[3] 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.

[4] R. G. Hautamäki, T. Kinnunen, V. Hautamäki, T. Leino, and A.-M. Laukkanen, “I-vectors meet imitators: on vulnerability of speaker verification systems against voice mimicry,” in *INTERSPEECH*, 2013.

[5] 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,” *Training*, vol. 10, no. 15, p. 3750, 2015.

[6] T. Kinnunen, M. Sahidullah, H. Delgado, M. Todisco, N. Evans, J. Yamagishi, and K. A. Lee, “The asvspoof 2017 challenge: Assessing the limits of replay spoofing attack detection.” [Online]. Available: http://www.spoofingchallenge.org/asvspoof2017overview.pdf

[7] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, “Large-scale video classification with convolutional neural networks,” in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2014, pp. 1725–1732.

[8] Y. Bengio, A. Courville, and P. Vincent, “Representation learning: A review and new perspectives,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 8, pp. 1798–1828, 2013.

[9] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[10] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, “Deepface: Closing the gap to human-level performance in face verification,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 1701–1708.

[11] C. Zhang, C. Yu, and J. H. L. Hansen, “An investigation of deep learning frameworks for speaker verification anti-spoofing,” *IEEE Journal of Selected Topics in Signal Processing*, no. 99, 2011.

[12] X. Tian, X. Xiao, C. E. Siong, and H. Li, “Spoofing speech detection using temporal convolutional neural network,” in *2016 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA)*, 2016.

[13] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, “Going deeper with convolutions,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1–9.

[14] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *arXiv preprint arXiv:1409.1556*, 2014.

[15] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun, “Overfeat: Integrated recognition, localization and detection using convolutional networks,” *arXiv preprint arXiv:1312.6229*, 2013.

[16] X. W. X., R. He, Z. Sun, and T. Tan, “A light cnn for deep face representation with noisy labels,” *IEEE Journal of Selected Topics in Signal Processing*, 2015.

[17] I. J. Goodfellow, D. Warde-Farley, M. Mirza, A. C. Courville, and Y. Bengio, “Maxout networks.” *ICML (3)*, vol. 28, pp. 1319–1327, 2013.

[18] M. Todisco, H. Delgado, and N. Evans, “A new feature for automatic speaker verification antispoofing: Constant q cepstral coefficients,” *Processings of Odyssey 2016*, 2011.

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

[20] N. Dehak, R. Dehak, P. Kenny, N. Brümmer, P. Ouellet, and P. Dumouchel, “Support vector machines versus fast scoring in the low-dimensional total variability space for speaker verification.” in *Interspeech*, vol. 9, 2009, pp. 1559–1562.

[21] M. Lin, Q. Chen, and S. Yan, “Network in network,” *arXiv preprint arXiv:1312.4400*, 2013.

[22] J. Chung, Ç. Gülçehre, K. Cho, and Y. Bengio, “Empirical evaluation of gated recurrent neural networks on sequence modeling,” *CoRR*, vol. abs/1412.3555, 2014. [Online]. Available: http://arxiv.org/abs/1412.3555

[23] D. P. and W. Ellis, “PLP and RASTA (and MFCC, and inversion) in Matlab,” 2005, online web resource. [Online]. Available: http://www.ee.columbia.edu/ dpwe/resources/matlab/rastamat/

[24] “Liblinear library.” [Online]. Available: http://www.csie.ntu.edu.tw/ cjlin/liblinear

[25] X. Glorot and Y. Bengio, “Understanding the difficulty of training deep feedforward neural networks.” in *Aistats*, vol. 9, 2010, pp. 249–256.

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

[27] “Bosaris toolkit.” [Online]. Available: https://sites.google.com/site/bosaristoolkit