INTERSPEECH 2019
September 15–19, 2019, Graz, Austria

Deep Residual Neural Networks for Audio Spoofing Detection

Moustafa Alzantot[1*], *Ziqi Wang*[2*], *Mani B. Srivastava*[1,2]

[1]Department of Computer Science, UCLA, Los Angeles, USA
[2]Department of Electrical and Computer Engineering, UCLA, Los Angeles, USA
malzantot@ucla.edu, wangzq312@ucla.edu, mbs@ucla.edu

Abstract

The state-of-art models for speech synthesis and voice conversion are capable of generating synthetic speech that is perceptually indistinguishable from bonafide human speech. These methods represent a threat to the automatic speaker verification (ASV) systems. Additionally, replay attacks where the attacker uses a speaker to replay a previously recorded genuine human speech are also possible. In this paper, we present our solution for the ASVSpoof2019 competition, which aims to develop countermeasure systems that distinguish between spoofing attacks and genuine speeches. Our model is inspired by the success of residual convolutional networks in many classification tasks. We build three variants of a residual convolutional neural network that accept different feature representations (MFCC, log-magnitude STFT, and CQCC) of input. We compare the performance achieved by our model variants and the competition baseline models. In the logical access scenario, the fusion of our models has zero t-DCF cost and zero equal error rate (EER), as evaluated on the development set. On the evaluation set, our model fusion improves the t-DCF and EER by 25% compared to the baseline algorithms. Against physical access replay attacks, our model fusion improves the baseline algorithms t-DCF and EER scores by 71% and 75% on the evaluation set, respectively.

Index Terms: ASVSpoof, Deep Learning, Spoofing Detection, Replay Attacks, Automatic Speaker Verification.

1. Introduction

Over the past decade, voice control has gained popularity as a practical and comfortable interface between users and smart devices. Due to the security and privacy sensitive nature of many applications (e.g., banking, health, and smart home) running on these devices, automatic speaker verification (ASV) [1] techniques have emerged as a form of biometric identification of the speaker. However, ASV systems are threatened by replay [2] and audio spoofing attacks where an attacker utilizes techniques such as voice conversion (VC) or speech synthesis (SS) to gain illegitimate control over user devices. Speech synthesis [3, 4, 5] and voice conversion [6, 7] have also progressed a lot over the past decade reaching the point where it has become very challenging to differentiate between their results and genuine users' speech. To enhance reliability against attacks, we combine ASV systems with audio spoofing detection systems that compute countermeasure scores to distinguish between spoofed and bonafide (genuine) speech. The automatic speaker verification spoofing and countermeasure challenge (ASVSpoof [1, 8, 2, 9]) competitions have emerged to assess the state-of-art methods for spoofing detection and promote further research in this critical challenge.

The first edition of the competition, ASVSpoof2015[8], focused on *logical access* scenarios where the attacker is using text-to-speech (TTS) [3, 7, 4] and voice conversion (VC) [6, 7] algorithms. The second edition of ASVSpoof competition, ASVSpoof2017 [2], focused on the *physical access* scenario where the attacker is performing *replay attack* by recording the genuine speech and then replay it to deceive the ASV system. The new edition of the competition, ASVSpoof2019 [9], extends the previous versions in several directions. First, it considers all three major forms of attacks: SS, VC, and replay attacks. Besides, the latest and strongest spoof algorithms are used to generate more natural counterexamples for spoof detection systems. Finally, while the previous competitions used the equal error rate (EER) as an evaluation metric, ASVSpoof 2019 adopts a newly proposed tandem decision cost function (t-DCF) as its primary metric and leaves EER as a secondary metric.

In this paper, we present our models submitted for the ASVSpoof2019 competition [9]. Inspired by the success of deep neural networks in many tasks [10, 11, 12], we pick a deep neural model as our model family. Among deep neural networks, convolutional networks have been the most successful in image classification [11], and have been recently applied to other data modalities such as Speech [13, 10], text [14] and ECG signals [15]. We consider different feature extraction algorithms to convert the input (raw time-domain speech waveform) into a 2D feature representation. That 2D feature representation is fed as an input into our convolutional model. A practical challenge in training very deep (consisting of many layers) convolutional networks is vanishing gradients that makes it hard for lower-layers (closer to input) to receive useful update signals during the training [16]. To overcome this issue, [16] recently proposed an effective solution called *residual networks* which employ skip connections that act as shortcuts allowing training updates to back-propagate faster towards the lower layers during training. Therefore, we also consider adding residual links to improve and stabilize the training of our models. A detailed description of our model architecture is provided in Section 3.2. Finally, we show how the fusion of countermeasure (CM) scores produced by models trained on different features help to increase the accuracy of the spoofing detection.

Our contribution in this paper is threefold. First, we design and implement a deep residual convolutional network to perform audio spoofing detection. Our models are released as open source[1]. Second, we provide a comparison between the performance of three different feature extraction algorithms (MFCC, log-magnitude STFT, and CQCC). Third, we evaluate the performance of our residual network with varying choices of input features against the two attack scenarios of ASVSpoof2019 (logical access, and physical access) using both the development (including only *known* attacks) and evaluation datasets (including both *known* and *unknown* attacks).

* First and second author contributed equally to this work.

[1]https://github.com/nesl/asvspoof2019

http://dx.doi.org/10.21437/Interspeech.2019-3174

The rest of this paper is organized as follows. Section 2 provides a summary of related work. Section 3.1 describes the feature extraction module of the system. Section 3.2 then describes our model architecture design and implementation. Section 4 includes our experiment results. Finally, Section 5 concludes the paper and points the future directions.

2. Related Work

While the participants of the previous ASVspoof2015 [8] have built several powerful solutions against audio spoofing, the state-of-the-art of audio spoofing techniques, e.g., TTS [3, 7] and VC [17], has also progressed a lot over the past four years. Likewise, this year's competition ASVSpoof2019 has a more realistic dataset for replay attacks compared to ASVSpoof2017 [2]. Prominent previous approaches against *logical access* attacks include [18] which used spectral-log-filter-bank and relative phase shift features as input to a model combining a deep neural network with support vector machine (SVM) classifier. [19] proposed using a DNN to compute a representative spoofing vector (s-vector). Then it uses normalized Mahalanobis distance between the s-vector and the class representative vector to calculate countermeasure scores. [20] uses relative phase information and group delay feature to train a Gaussian Mixture Model (GMM) for detecting spoofing attacks. Against *replay* attacks, [21] have previously developed a deep learning model combining both CNN and RNN that lead to 6.73% EER on the ASVSpoof2017 evaluation dataset. In ASVSpoof2017, [22] also used a residual convolutional network, but with different an architecture and input features, to obtain 13.44% EER on the eval set.

3. Model Design

The goal of ASVspoof challenge is to compute a countermeasure (CM) score for each input audio file. A high CM score indicates a bonafide speech, and a low CM score indicates a spoofing attack. We created a deep residual network that performs binary classification. To prepare the features as the convolutional network inputs, we process the raw audio waveform first a by a feature extraction step which we will discuss in the next section.

3.1. Feature Extraction

We prepare features from raw audio waveform by one of the following feature extraction algorithms: the Mel-Frequency Cepstral Coefficients (MFCCs), the Constant Q Cepstral Coefficients(CQCCs), and the Logarithmic Magnitude of Short-Time Fourier Transform(log-magnitude STFT).

Mel-frequency Cepstral Coefficients (MFCCs): MFCC is a widely used feature for speech recognition and other applications like music genre classification. MFCC is achieved by computing the short-time-Fourier-transform (STFT), then mapping the spectrum into Mel-Spectrum through a filter bank, and finally calculating a discrete cosine transform(DCT). We pick the first 24 coefficients. We also find the performance can be improved if we concatenate the MFCC with its first-order $\Delta MFCC$ and second derivative $\Delta^2 MFCC$ to produce our feature representation which is a 2D matrix whose x axis is the time and y axis is the 72 elements of $(MFCC, \Delta MFCC, \Delta^2 MFCC)$. This improvement is because derivatives of MFCC capture the dynamics in cepstral coefficients.

Constant Q Cepstral Coefficients(CQCCs): Instead of using STFT, the CQCC uses constant-Q transform(CQT) which was initially proposed for music processing. While STFT imposes a regularly spaced frequency bins, the CQT uses geometrically spaced frequency bins. Thus, it offers a higher frequency resolution at lower frequencies and higher temporal resolution at higher frequencies. To compute CQCC, after applying CQT, we calculate a power spectrum and take a logarithm. Then a *uniform re-sampling* is performed, followed by a DCT to get the CQCCs(which is also a 2D matrix). More details of CQCC can be found in [23].

Logarithmic Magnitude of STFT: An advantage of deep learning models is their capabilities of representation learning [24, 25] by automatically learning high-level features from raw input data. This ability has led to neural models which process raw input images to outperform models dealing with human-engineered features. Inspired by this, we also train models with the log magnitude of STFT as the input. We first compute the STFT on hamming windows (window size = 2048 with 25% overlap). Then we calculate the magnitude of each component and convert it to log scale. The output matrix captures the time-frequency characteristics of the input audio waveform and is fed directly as an input to our neural model without any further transformations or conversions. While this input representation is rawer than either MFCC or CQCC, we rely on the representation learning abilities of neural networks to transform this input into higher-level representations within the hidden layers of our model.

3.2. Model Architecture

We build three different models variants `MFCC-ResNet`, `CQCC-ResNet`, and `Spec-ResNet` which process MFCC, CQCC and log-magnitude STFT (which turns out to be a spectrogram) input features, respectively. The three variants have a nearly identical architecture, but they differ from each other in the input shape to accommodate the differences in the dimensions of input features, and consequentially also the number of units in the first fully connected layer which is after the last residual block, as we will explain later.

Figure 1 shows the architecture of the `Spec-ResNet` model which takes the log-magnitude STFT as input features. First, the input is treated as a single channel image and passed through a 2D convolution layer with 32 filters, where filter size = 3×3, stride length = 1 and padding = 1. The output volume of the first convolution layer has 32 channels and is passed through a sequence of 6 *residual blocks*. The output from the last residual block is fed into a dropout layer [26] (with dropout rate = 50%) followed by a hidden fully connected (FC) layer with leaky-ReLU [27] activation function ($\alpha = 0.01$). Outputs from the hidden FC layer are fed into another FC layer with two units that produce classification logits. The logits are finally converted into a probability distribution using a final `softmax` layer.

The structure of a residual block is shown in Figure 2. Each residual block has a `Conv2D` layer (32 filters, filter size = 3×3, stride = 1, padding = 1) followed by a batch normalization layer [28], a leaky-ReLU activation layer [27], a dropout (with dropout probability = 0.5) [26], and another final `Conv2D` layer (also 32 filters and filter size = 3×3, but with stride = 3 and padding = 1). Dropout is used as a regularizer to reduce the model overfitting, and batch normalization [28] accelerates the network training progress. A skip-through connection is established by directly add the inputs to the outputs. To guarantee that the dimension agrees, we apply a `Conv2D` layer (32 fil-

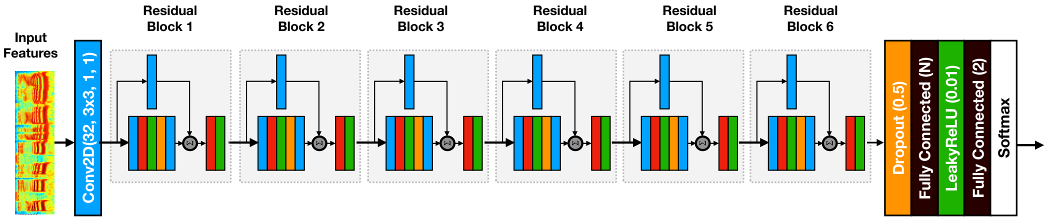

Figure 1: *Model architecture for the Spec-ResNet model. Detailed structure of residual blocks is shown in 2.*

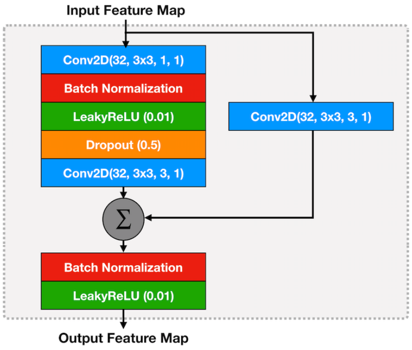

Figure 2: *Detailed architecture of the convolution block with residual connection.*

ters, filter size = 3×3, stride = 3, padding = 1) on the bypass route. Finally, batch normalization [28] and leaky-ReLU non-linearity are used to produce the residual block output.

All models are trained by minimizing a weighted cross-entropy loss function where the ratio of between weights assigned to genuine and spoofed examples are 9:1, in order to mitigate the imbalance in the training data distribution. The cost function is minimized using Adam optimizer [29] with learning rate = 5×10^{-5} for 200 epochs with batch size = 32. After each epoch we save the model parameters, and we finally use the parameters with the best performance on the validation dataset.

The final countermeasure score (CM) is computed from the softmax outputs using the log-likelihood ratio.

$$CM(s) = \log(p(\text{bona fide}|s;\theta)) - \log(p(\text{spoof}|s;\theta))$$

where s is the given audio file and θ represents the model parameters.

4. Evaluation

We implemented our neural network model using PyTorch [30] and trained our models using a desktop machine with TitanX GPU. Feature extraction was done using the `librosa` [31] python library.[2].

4.1. Dataset and Baseline Models

The competition organizers provide a dataset of non-overlapping short audio files for each competition track. The bonafide voice clips come from 78 human (male and female) speakers. The dataset is divided into three partitions with disjoint sets of speakers: *training* (8 male, 12 female), *development* (4 male, 6 female), and *evaluation* (21 male, 27 female). The spoofed audio in the *logical access* scenario is generated using 17 different speech synthesis and voice conversion toolkits. Six of these attack types are considered *known* attacks and are used to generate the training and development datasets while the other 11 attacks are considered *unknown* and are used, along with two of the *known* attacks, to generate the evaluation dataset. For *physical access* scenario, replay attacks are recorded and replayed in the 27 different acoustic configurations and nine different settings (combinations of three categories of recording distance and three levels of replay device quality) [9]. Evaluation data for the *physical access* are generated from different impulse responses and therefore represents *unknown* attacks.

Baseline Models : For each track of the competition, the organizers have provided implementations for two baseline models which are using Gaussian mixture models (GMMs) [32, 33] using the Linear Frequency Cepstral Coefficients (LFCC) and CQCC features.

4.2. Evaluation Metrics

The evaluation scores are computed using the following metrics on both the development dataset (*known* attacks) and evaluation dataset (both *known* and *unknown* attacks):

t-DCF [34]: the *tandem detection cost function* is the new primary metric in the ASVSpoof 2019 challenge. It was proposed as a reliable scoring metric to evaluate the combined performance of ASV and CMs.

EER: the Equal Error Rate is used as a secondary metric. EER is determined by the point at which the miss (false negative) rate and false alarm (false positive) rate are equal to each other.

4.3. Results

Table 1 shows a comparison between the scores of our three model variants (`MFCC-Resnet`, `Spec-ResNet`, `CQCC-ResNet`) and the baseline algorithms (`LFCC-GMM`, and `CQCC-GMM`) on both the development and evaluation datatsets. *Fusion* represents the result of doing *weighted* average of the individual ResNet models' *CM* scores to provide a final *CM* score, where fusion weights are assigned based on the single model's performance on the validation dataset.

4.3.1. Logical Access Results

As shown in Table 1, Our `Spec-ResNet` and `CQCC-ResNet` have a significantly smaller t-DCF and EER scores than the baseline algorithms on the development set (*known* attacks) of the logical access scenario. The fusion of the models achieves a *perfect* score of zero EER and t-DCF on the development set. However, in the evaluation set results, our models outperform the baseline models only in the EER of `CQCC-ResNet` and t-DCF score of `MFCC-ResNet`. This highlights the difficulty of generalizing a spoofing detection system to *unknown* attack algorithms. Nevertheless, our model fusion shows t-DCF $= 0.1569$ and EER $= 6.02$ which are approximately a 25% improvement over the best scores of baseline algorithms.

To provide a better analysis of the performance of our

[2]For the CQCC for which we used the MatLab code provided by competition organizers

Table 1: *t-DCF and EER scores for the different models as measured on the development and evaluation sets for both logical and physical access scenarios.*

	Logical Access				Physical Access			
	Development		Evaluation		Development		Evaluation	
Model	t-DCF	EER	t-DCF	EER	t-DCF	EER	t-DCF	EER
Baseline LFCC-GMM	0.0663	2.71	0.2116	8.09	0.2554	11.96	0.3017	13.54
Baseline CQCC-GMM	0.0123	0.43	0.2366	9.57	0.1953	9.87	0.2454	11.04
MFCC-ResNet	0.1013	3.34	**0.2042**	9.33	0.3770	15.91	-	-
Spec-ResNet	0.0023	0.11	0.2741	9.68	0.0960	3.85	**0.0994**	**3.81**
CQCC-ResNet	0.0002	0.01	0.2166	**7.69**	0.1026	4.30	0.1070	4.43
Fusion	0.0000	0.00	**0.1569**	**6.02**	0.0581	2.65	**0.0693**	**2.78**

model against both *known* and *unknown* attacks, the t-DCF scores of our models against each attack type are shown in Figure 3. Attacks from A01 to A06 are *known* attacks (from the development set) while attacks from A07 to A19 are the 11 *unknown* and two *known* attacks (from the evaluation set). From Figure 3, we can see that our models still work well against most attack types except for only two types of the *unknown* attacks, namely A17 and A18. Both A17 and A18 are voice conversion algorithms, where A17 is based on waveform filtering and A18 is based on vocoders. In comparison to the baseline models, the `CQCC-GMM` model also perform poorly on A17(t-DCF=0.9820), which suggest that CQCC is easier to be deceived by waveform filtering based video conversion attacks. Both the `CQCC-GMM` and `LFCC-GMM` work fine on A18, so it is possible that ResNet is more vulnerable to vocoder based video conversion attacks.

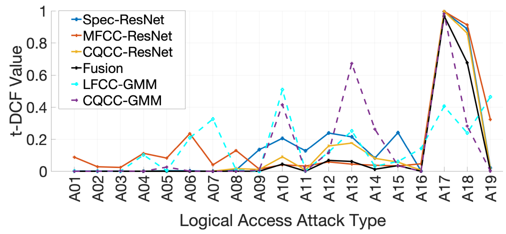

Figure 3: *t-DCF scores of different models against different attack types (both TTS and VC) in the logical access scenario.*

4.3.2. *Physical Access Results*

In the physical access scenario, both `Spec-ResNet` and `CQCC-ResNet` have significantly improved both the EER and t-DCF. As shown in Table 1, our best single model (`Spec-ResNet`) is 50% and 60% better than the best baseline results according to the development set EER and t-DCF, respectively. According to the evaluation set scores, `Spec-ResNet` reduces the t-DCF and EER of baseline algorithms by 60% and 65%, respectively. Furthermore, the fusion of our models leads to 71% and 75% improvement.

Table 2 provides detailed results of model performance over different replay attack settings. Each setting is named with two letters. The first letter stands for the distance of the recording device from the bona-fide speaker. 'A' means 10-50 cm, 'B' means 50-100 cm, and 'C' means >100cm. The second letter indicates the quality of replay devices, where A means perfect, B means high, and C means low. From the results it is easy to see that, as the distance decreasing and recording device getting better, the anti-spoof task becomes more and more difficult. The worst results are achieved at setting 'AA'. Another thing to notice is that, while `Spec-ResNet` is generally performing better than `CQCC-ResNet` while in some cases like 'BB', 'BC', 'CB', and 'CC', `CQCC-ResNet` outperforms `Spec-ResNet`.

Generally, the system performs better on physical access scenarios that on logical access. This is probably caused by the challenge of generalization, as in logical access, most attacks in the testing dataset are diverse and unknown, while in physical access the features come from the replay channel properties and are easier to learn and generalize.

Table 2: *Detailed comparison between the two best single models and the fusion model in Physical Access scenario under different replay attack settings.*

Attack Type	**CQCC-ResNet**		**Spec-ResNet**		**Fusion**	
	t-DCF	EER	t-DCF	EER	t-DCF	EER
AA	0.2857	10.59	**0.2473**	**9.17**	0.1845	6.78
AB	0.0690	2.57	**0.0638**	**2.22**	0.0468	1.77
AC	0.0464	1.75	**0.0436**	**1.56**	0.0219	0.80
BA	0.1404	5.46	**0.1300**	**4.82**	0.0855	3.29
BB	**0.0295**	**1.18**	0.0374	1.34	0.0230	0.79
BC	**0.0213**	**0.84**	0.0240	0.86	0.0086	0.36
CA	0.1173	4.55	**0.1105**	**4.01**	0.0705	2.71
CB	**0.0266**	**1.00**	0.0342	1.19	0.0171	0.59
CC	**0.0209**	**0.82**	0.0254	0.87	0.0074	0.28

5. Conclusions

In this paper, we presented a novel audio spoofing detection system for both logical access and physical access scenarios. We provide comparisons between the performance of our model combined with three feature different feature extraction algorithms. According to the evaluation dataset scores, against replay attacks, the fusion of our models CM scores improves the t-DCF and EER metrics of baseline algorithm by 71% and 75% respectively. Also, against the TTS and VC attacks, our fusion of models improves the t-DCF and EER metrics by approximately 25% each. Our future work is to study how to improve the generalization of our model against *unknown* attacks. One possible solution is to employ advanced fusion to build a 'wide-and-deep' network as proposed in [35]. The key idea of this new proposal is to concatenate the features from each model's last fully connected layers and use a shared softmax layer as the output layer. This might be able to train the networks to collaborate with each other and achieve a better fusion result.

Acknowledgement. This research was supported in part by the U.S. Army Research Laboratory and the UK Ministry of Defence under Agreement Number W911NF-16-3-0001; by the CONIX Research Center, one of six centers in JUMP, a Semiconductor Research Corporation (SRC) program sponsored by DARPA; and, by the National Science Foundation under award # CNS-1705135. Any findings in this material are those of the author(s) and do not reflect the views of any of the above funding agencies. The U.S. and U.K. Governments are authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

6. References

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

[2] T. Kinnunen, N. Evans, J. Yamagishi, K. A. Lee, M. Sahidullah, M. Todisco, and H. Delgado, "Asvspoof 2017: Automatic speaker verification spoofing and countermeasures challenge evaluation plan," *Training*, vol. 10, no. 1508, p. 1508, 2017.

[3] 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.

[4] Z. Wu, O. Watts, and S. King, "Merlin: An open source neural network speech synthesis system." in *SSW*, 2016, pp. 202–207.

[5] L. Juvela, B. Bollepalli, X. Wang, H. Kameoka, M. Airaksinen, J. Yamagishi, and P. Alku, "Speech waveform synthesis from mfcc sequences with generative adversarial networks," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 5679–5683.

[6] T. Toda, L.-H. Chen, D. Saito, F. Villavicencio, M. Wester, Z. Wu, and J. Yamagishi, "The voice conversion challenge 2016." in *Interspeech*, 2016, pp. 1632–1636.

[7] W.-C. Huang, C.-C. Lo, H.-T. Hwang, Y. Tsao, and H.-M. Wang, "Wavenet vocoder and its applications in voice conversion," *ROCLING 2018*, p. 96, 2018.

[8] 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 *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

[9] A. consortium, "Asvspoof 2019: Automatic speaker verification spoofing and countermeasures challenge evaluation plan," 2019.

[10] D. Amodei, S. Ananthanarayanan, R. Anubhai, J. Bai, E. Battenberg, C. Case, J. Casper, B. Catanzaro, Q. Cheng, G. Chen *et al.*, "Deep speech 2: End-to-end speech recognition in english and mandarin," in *International conference on machine learning*, 2016, pp. 173–182.

[11] 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.

[12] A. Esteva, B. Kuprel, R. A. Novoa, J. Ko, S. M. Swetter, H. M. Blau, and S. Thrun, "Dermatologist-level classification of skin cancer with deep neural networks," *Nature*, vol. 542, no. 7639, p. 115, 2017.

[13] O. Abdel-Hamid, A.-r. Mohamed, H. Jiang, L. Deng, G. Penn, and D. Yu, "Convolutional neural networks for speech recognition," *IEEE/ACM Transactions on audio, speech, and language processing*, vol. 22, no. 10, pp. 1533–1545, 2014.

[14] X. Zhang, J. Zhao, and Y. LeCun, "Character-level convolutional networks for text classification," in *Advances in neural information processing systems*, 2015, pp. 649–657.

[15] P. Rajpurkar, A. Y. Hannun, M. Haghpanahi, C. Bourn, and A. Y. Ng, "Cardiologist-level arrhythmia detection with convolutional neural networks," *arXiv preprint arXiv:1707.01836*, 2017.

[16] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[17] J. Lorenzo-Trueba, J. Yamagishi, T. Toda, D. Saito, F. Villavicencio, and Z. Kinnunen, Tomi a Ling, "The voice conversion challenge 2018: Promoting development of parallel and nonparallel methods," *arXiv preprint arXiv:1804.04262*, 2018.

[18] J. Villalba, A. Miguel, A. Ortega, and E. Lleida, "Spoofing detection with dnn and one-class svm for the asvspoof 2015 challenge," in *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

[19] N. Chen, Y. Qian, H. Dinkel, B. Chen, and K. Yu, "Robust deep feature for spoofing detectionthe sjtu system for asvspoof 2015 challenge," in *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

[20] L. Wang, Y. Yoshida, Y. Kawakami, and S. Nakagawa, "Relative phase information for detecting human speech and spoofed speech," in *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

[21] G. Lavrentyeva, S. Novoselov, E. Malykh, A. Kozlov, O. Kudashev, and V. Shchemelinin, "Audio replay attack detection with deep learning frameworks." in *Interspeech*, 2017, pp. 82–86.

[22] Z. Chen, Z. Xie, W. Zhang, and X. Xu, "Resnet and model fusion for automatic spoofing detection." in *INTERSPEECH*, 2017, pp. 102–106.

[23] M. Todisco, H. Delgado, and N. Evans, "Constant q cepstral coefficients: A spoofing countermeasure for automatic speaker verification," *Computer Speech & Language*, vol. 45, pp. 516–535, 2017.

[24] 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.

[25] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

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

[27] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1026–1034.

[28] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.

[29] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[30] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," 2017.

[31] B. McFee, C. Raffel, D. Liang, D. P. Ellis, M. McVicar, E. Battenberg, and O. Nieto, "librosa: Audio and music signal analysis in python," in *Proceedings of the 14th python in science conference*, 2015, pp. 18–25.

[32] D. A. Reynolds and R. C. Rose, "Robust text-independent speaker identification using gaussian mixture speaker models," *IEEE transactions on speech and audio processing*, vol. 3, no. 1, pp. 72–83, 1995.

[33] D. A. Reynolds, T. F. Quatieri, and R. B. Dunn, "Speaker verification using adapted gaussian mixture models," *Digital signal processing*, vol. 10, no. 1-3, pp. 19–41, 2000.

[34] T. Kinnunen, K. A. Lee, H. Delgado, N. Evans, M. Todisco, M. Sahidullah, J. Yamagishi, and D. A. Reynolds, "t-dcf: a detection cost function for the tandem assessment of spoofing countermeasures and automatic speaker verification," *arXiv preprint arXiv:1804.09618*, 2018.

[35] H.-T. Cheng, L. Koc, J. Harmsen, T. Shaked, T. Chandra, H. Aradhye, G. Anderson, G. Corrado, W. Chai, M. Ispir *et al.*, "Wide & deep learning for recommender systems," in *Proceedings of the 1st workshop on deep learning for recommender systems*. ACM, 2016, pp. 7–10.