



# Attacker Attribution of Audio Deepfakes

Nicolas M. Müller<sup>1\*</sup>, Franziska Dieckmann<sup>2\*</sup>, Jennifer Williams<sup>3</sup>

<sup>1</sup>Fraunhofer AISEC, Germany

<sup>2</sup>Technical University of Munich, Germany

<sup>3</sup>University of Southampton, UK

nicolas.mueller@aisec.fraunhofer.de, franziska.dieckmann@tum.de, j.williams@soton.ac.uk

## Abstract

Deepfakes are synthetically generated media often devised with malicious intent. They have become increasingly more convincing with large training datasets advanced neural networks. These fakes are readily being misused for slander, misinformation and fraud. For this reason, intensive research for developing countermeasures is also expanding. However, recent work is almost exclusively limited to deepfake detection - predicting if audio is real or fake. This is despite the fact that attribution (*who created which fake?*) is an essential building block of a larger defense strategy, as practiced in the field of cybersecurity for a long time. This paper considers the problem of deepfake attacker attribution in the domain of audio. We present several methods for creating *attacker signatures* using low-level acoustic descriptors and machine learning embeddings. We show that speech signal features are inadequate for characterizing attacker signatures. However, we also demonstrate that embeddings from a recurrent neural network can successfully characterize attacks from both known and unknown attackers. Our attack signature embeddings result in distinct clusters, both for seen and unseen audio deepfakes. We show that these embeddings can be used in downstream-tasks to high-effect, scoring 97.10% accuracy in attacker-id classification.

**Index Terms:** audio spoofing, deepfake, clustering, embedding

## 1. Introduction

Deepfakes are synthetically generated media content often created and delivered with malicious intent. Very large datasets, increasing compute power, and the use of advanced neural networks have made it possible to create very convincing deepfakes. Deepfakes are already being misused in everyday life for slander [1], fake news [2] and even financial fraud [3]. For this reason, the detection of deepfakes (or spoofs) is a major subject of current research, both in the domain of video and audio. Especially in the domain of audio, most research is currently focused only on the *detection* of fake voice recordings. For example, the popular ASVspoof Challenge [4], only considers detection tasks for audio deepfakes but does not handle attacker *attribution*. Attribution is more challenging than detection since the latter is essentially a binary classification problem (i.e., speech is either real or fake), but attribution requires a much finer gradation than ‘real or fake’. In this paper we present work on *attacker attribution* using a technique that allows us to develop special signatures that represent particular attackers.

Working on attacker attribution is important because it allows a better understanding of the threat landscape. In cybersecurity, for example, attribution of attacks is already an important part of threat mitigation strategy [5–9]. Our paper deals with the

attribution of audio deepfakes through the creation of attacker signatures and we make three main contributions. (1) We introduce and evaluate a signature based on 16 low-level acoustic features such as pitch, jitter, etc. with respect to how well the signature performs in characterising audio deepfake attackers. We show that these simpler features have only limited applicability. (2) We show that neural embedding signatures can successfully cluster deepfake speech audio with respect to attacker ID from a labeled corpus. We evaluate the neural embeddings on a large audio deepfake corpus and show that the resulting embeddings allow simple downstream models to achieve high classification accuracy.

## 2. Related Work

The term deepfake originated in 2017 from a Reddit user who posted face-swapped pornographic videos [10]. Since then, media content forgery using machine learning has developed rapidly which in turn motivated a flurry of research on deepfake detection [11, 12] and corresponding challenges such as Facebook’s Deepfake Detection Challenge [13]. In the domain of audio and voice, which is the focus of this paper, these fakes are commonly referred to as ‘spoofs’ [4].

Audio spoofs can be created in several ways, and one such way involves using *text-to-speech* (TTS) synthesis models such as Tacotron [14] combined with a vocoder such as Griffin-Lim [15] or *neural vocoders* such as WaveNet [16]. With either of these vocoder methods, the spectrogram is inverted and a raw waveform of speech can be obtained. Tacotron and its successor Tacotron 2 [17] have triggered considerable follow-up research, which optimizes either synthesis quality [18], minimizes inference time [19] or allows for controllable prosody [20]. However, TTS synthesis technology can also be used to create deepfakes of a voice by cloning a person’s voice and controlling what they say without consent.

The threat that deepfakes pose to society has triggered research in the field of audio spoof detection and led to the establishment of a large biennial challenge called the “Automatic Speaker Verification and Spoofing Countermeasures” (ASVspoof) series [4]. In this challenge, the organizers provide large datasets which contain both authentic speech (i.e. spoken by a real human) and spoofed speech (i.e. fake speech from TTS synthesis, voice conversion, or neural vocoder systems). To take the progress of TTS synthesis and voice conversion into account, new benchmark datasets are released every other year (c.f. 4.1). This challenge has incited many papers which have advanced state-of-the-art in audio spoof detection [21–24]. However, the community has perhaps been too optimistic about the detection rate of new audio spoofs not included in the training dataset, as it has been shown that some datasets contain artifacts that make classification tasks trivial [25] due to the amount

\*Equal contribution

of silence padding in a waveform.

Recording environment signatures were first created for modeling physical attack characteristics in the ASVspoof 2019 Challenge [26]. While the environment embeddings did help with attack detection, they did not identify particular types of attacks. A related problem is attacker attribution, i.e. answering the question: 'Who created this audio spoof?'. Attribution requires the creation of attacker signatures which remain constant over all of the attacker's audio spoofs. In this case, an attacker is identified by the specific speech synthesis setup they use which includes not only the choice of architecture but training data and hyperparameters as well. In the domain of video deepfakes, there is very little prior work on attribution [27, 28]. In the domain of audio deepfakes the work of [29] uses clustering [29] for spoof *detection*, but there is no prior work on attacker attribution.

### 3. Attacker Signatures

Recall that learning attacker signatures is a different technical problem than detecting deepfakes. The primary difference involves the assumption that the audio is already known to be a deepfake and therefore the task is to attribute each deepfake to the system of origin (vocoder, TTS system, voice conversion system, etc). This section describes two main approaches for creating attacker signatures: extracting low-level signal features and learning embeddings from a neural network.

We define the attacker signatures as follows: For attackers  $A_0, A_1, \dots$  and raw audio waveforms  $W_0, W_1, \dots$ , we are interested in a signature  $f$  s.t.  $f(W_l) = f(W_k)$  if and only if  $W_l, W_k$  originate from the same attacker. We can relax the problem and write  $f(W_l) \approx f(W_k)$ . This requires a notion of distance in  $f(W)$ -space, for example  $L_2$  or cosine similarity. Note that we use the notation of  $A_0$  to indicate bonafide speech (i.e. authentic, non-spoofed audio waveforms) since this notation follows a similar pattern as the labeled ASVspoof dataset that we used in our experiments.

#### 3.1. Low-Level Features

A natural first approach for creating attacker signatures is using features that can be computed directly from raw audio. Related work has shown that even simple features, such as the length of audio silence, can suffice to verify authenticity [25]. We extract the following low-level signal features from raw audio using *Parselmouth* [30] with *Praat* [31] and *Pydub* [32]. Some of our features are similar to what could be obtained through software toolkits like OpenSMILE [33]. The extracted features are combined into 16-dimensional vectors and used in our analysis for clustering and classifying attackers.

**Fundamental frequency.** The mean, minimum and maximum F0 (as extracted by *Parselmouth*), including the standard deviation and mean absolute slope (MAS). The pitch MAS is the frequency of vibration of the sound waves. All pitch values were calculated from the utterance-level waveform. This feature was chosen because it is well-known that machine-generated speech often lacks naturalness for prosody with variation among different types of systems [34].

**Distortion and shimmer.** We used jitter (i.e. time distortions from the digital audio signal), and shimmer, where shimmer is the average absolute difference between the amplitudes of consecutive periods, divided by the average amplitude [35]. These two types of features are interesting because they can capture instant-to-instant changes in frequency and amplitude

of the signal, and are known to capture some differences between biological systems and machine systems [34].

**Speaker gender.** This is a binary feature for male vs. female gender of the speaker in the audio file. While this feature may seem naive, TTS pre-training or warm-up is often based on single-speaker data such as the highly popular female speaker in the LJSpeech Dataset [36]. The use of single-speaker data for pre-training could potentially cause bias in speech synthesis.

**Duration and loudness.** The duration feature that we used was simply the duration of the audio signal on a per-file basis. The loudness in dBFS (db relative to the maximum possible loudness). A square wave at maximum amplitude will be roughly 0 dBFS (maximum loudness) [37].

**Signal amplitude, power, and energy.** For amplitude, we used the highest amplitude of the signal, with and without the conversion into dBFS (which specifies the value relative to the highest possible amplitude). We also calculated the power and energy over the entire audio file.

**Noise ratio.** The mean and standard deviation of a short-term Harmonic to Noise Ratio (HNR) analysis. HNR is known to be predictive of audio deepfakes [38].

#### 3.2. Neural Embeddings

Alternatively, we can learn the attacker signature. We design a neural network  $f_\theta$  which maps a waveform  $W$  to an attacker signature  $f_\theta(W)$ . Following the architecture presented in [39], this network consists of a stack of three recurrent layers. Each layer in the stack consists of an LSTM with 768 neurons, whose output is fed into a 256-dimensional dense layer (called projection layer). The output of the network is an embedding in  $f_\theta(W) \in \mathbb{R}^{256}$ . The model is trained via the Angular Prototypical loss [40], which is a form of cosine similarity loss.

## 4. Experimental Results

#### 4.1. Evaluation Data

We use the ASVspoof 2019 dataset [4] to evaluate our attacker signatures that were proposed in Section 3. Specifically, we use the *Logical Access* (LA) part of ASVspoof 2019, which we abbreviate in this paper as: ASV19. It consists of speech audio files which are either bonafide (i.e. authentic recordings of human speech) or spoofed audio (i.e. synthesized or fake audio). The fake audio originates from 19 different attackers, labeled  $A_1 - A_{19}$ . For each attacker, there are 4914 audio recordings, while there are 7355 bonafide samples. ASV19 was the official dataset for the ASVspoof 2019 challenge [4], which focused on spoof detection, i.e. binary classification of authenticity. A significant number of related work have used ASV19 data to develop detection algorithms [21–24].

#### 4.2. Evaluation Metric

To evaluate the quality of our clustering given a labeled dataset, we compute the average class-conditional variance<sup>1</sup> as follows in Equation 1 and Equation 2:

$$var_C(X) = \frac{1}{|C|} \sum_{c \in C} var(X, y = c) \quad (1)$$

where

$$var(X, y = c) = \frac{1}{N_c - 1} \sum_{i=1}^{N_c} (\bar{x}_c - x_c^{(i)})^2 \quad (2)$$

<sup>1</sup>we estimate both mean and variance from a sample distribution and thus use Bessel-correction  $N_c - 1$  in the denominator.

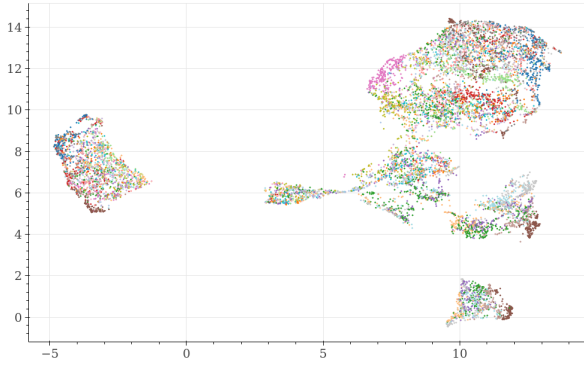


Figure 1: Clusters formed by our 16-dimensional embeddings from low-level signal features, using a 2D UMAP projection of the embedding space. The color of the data points indicates the different attacker ID labels.

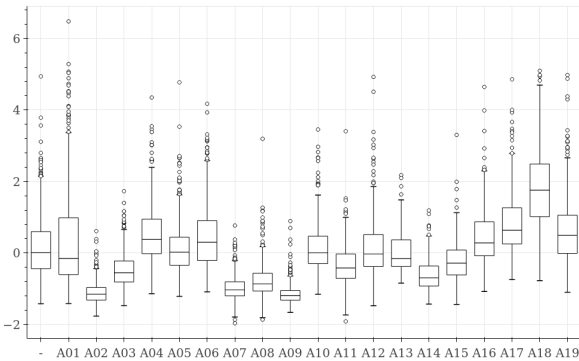


Figure 2: The values of the jitter static features, shown as box plot per attack ID. While it is somewhat indicative of the class, the domain overlaps for many classes, which limits its suitability for attacker signature creation.

describes the class-conditional variance and  $\bar{x}_c = \sum_{i=1}^{N_c} x_c^{(i)}$  is the centroid of class  $c$ ; with  $N_c$  instances  $x_c^{(i)}$  belonging to class  $c$ . Intuitively, this describes how closely packed the clusters are: the smaller  $var_C$ , the better the clustering w.r.t. to the target classes  $C$ . Note that this metric is sensitive to the scale of the data, which is why we consistently apply standard-normalization first. Lower values for class-conditional variance are indicative of better clustering.

### 4.3. Evaluation of Low-Level Signal Features

Figure 1 shows how our 16-dimensional embeddings form clusters when projected into a 2D space using UMAP [41]. The attacker ID labels are indicated by the colors of the data points. While some of the attack types are grouped together, there is no clear inter-label separation, and the clusters comprise many different attack IDs. While the data seems to cluster locally, there is no strong separation between the clusters. Rather, single clusters comprise multiple labels. Thus the conventional features are limited in their applicability to attacker signature identification.

We can verify this numerically by computing the average distance of each instance to its class centroid, as shown in Table 1. For reference, we have also included a box-plot of our strongest low-level signal feature, *jitter*, which achieves a

Low-Level Signal Feature	Class-Conditional Variance (Avg.)
duration	0.97
energy	0.93
gender	0.69
HNR mean	0.61
HNR std	0.71
<b>jitter</b>	<b>0.53</b>
loudness	0.67
max loudness	1.21
peak amplitude	1.21
pitch MAS	0.72
pitch max	0.85
pitch mean	0.98
pitch min	0.87
pitch std	0.80
power	0.73
shimmer	0.73
all	0.83

Table 1: Evaluating the resulting clusters from our 16-dimensional embeddings created from low-level signal features. The average class-conditional variance is reported as averaged over all attack IDs. Lower values would indicate that a feature contributes to better clustering of attack IDs.

conditional-class variance of 0.53. The jitter feature is somewhat indicative of attacker ID, as shown in Figure 2. To compute the values reported in Table 1, we first compute the class-conditional variance and then compute the average over all attack IDs. All features are standard-normalized. We see that the class-conditional variance is high. The usability of these features for attack signature clustering is very limited and the class-conditional values explain why clustering was not achieved in Figure 1. The corresponding boxplot in Figure 2 shows that while *jitter* is somewhat indicative of attacker ID, the ranges of values for jitter overlap significantly. Of all the low-level features, jitter was the best one based on our analysis. We conclude that our selected low-level features are of limited usefulness. Note that the majority of the features in Table 1 are even less useful than jitter. Overall, the class conditional variance is 0.83 on average, which is much worse than jitter at 0.53.

Another method that we use to evaluate the features is through supervised classification. We train a classifier to predict the attacker ID, given the 16-dimensional feature vector. We choose a simple feed-forward network with 3 hidden layers, 50 neurons per layer, ReLU activation [42], and learning rate of 0.001. We train on 90% of the ASV19 data while evaluating on the other 10% as held-out test data. We achieve an accuracy of 56.96%, which is significantly better than random guessing (the random baseline has 5% accuracy, given 20 different attacker IDs), but inadequate still. We conclude that the low-level features from Section 3.1 are not suitable for attacker signatures.

### 4.4. Evaluation of Neural Embeddings

We present our evaluation of neural embeddings for attacker signature (c.f. Section 3.2) evaluated on the ASV19 dataset. We evaluate two different scenarios: *in-domain* and *out-of-domain*. We describe each scenario and present clustering results for each one. We also describe how well the attack signature embeddings perform in a separate classification task. For the *in-domain* scenario, we train a neural network to learn embeddings

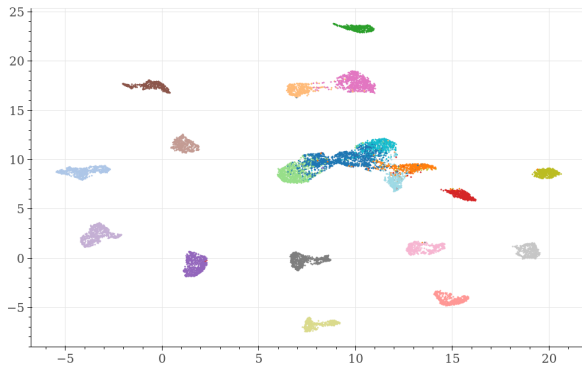


Figure 3: *In-domain evaluation of our neural attack signatures on ASV19. All attacker types were seen during training.*

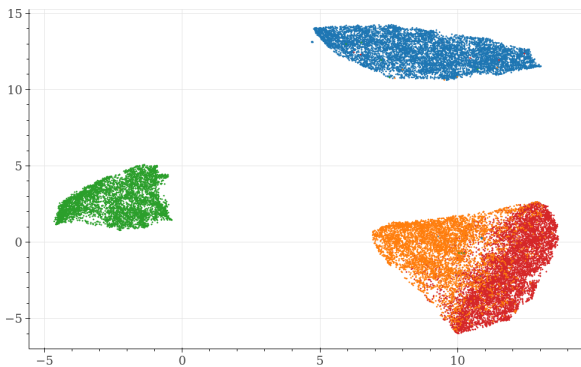


Figure 4: *Our-of-domain evaluation of our neural attack signatures on ASV19. Four attackers were hld-out from training.*

using the training partition of the ASV19 dataset. Our network is trained on 90% of this partition and evaluated on the remaining 10%. By treating this 10% as our test data, we are able to evaluate our embeddings using new audio recordings from attackers seen during training. For the *out-of-domain* scenario, we select four randomly chosen attacker IDs (A02, A04, A012, A14) and reserve all of their audio samples as a test set. We then train on the remainder of the data. This allows us to evaluate our ability to attribute unseen audio recordings from unknown attackers, as we would expect in a real-world application such as a tool for cybersecurity purposes.

The resulting clusterings for each scenario are shown in Figure 3 and Figure 4. As expected, we observe that the *in-domain* task seems easier than the *out-of-domain* task due to the successful separation of the large number of clusters. Though in both cases we obtain sensible clustering. Computation of the average class-conditional variance yields 0.19 for the *in-domain* task and 0.46 for the *out-of-domain* task. Both of these significantly outperform the clustering evaluation from low-level signal features, which had an average class-conditional variance of 0.83. The out-of-domain scenario is more challenging, since the audio file content and attackers are unseen during training. Nevertheless, we obtain very clear clustering, indicating that our neural embedding signatures can generalize beyond known attacker IDs and may be usable in a real-world scenario.

Additionally, we evaluate the neural embeddings as a downstream classification task. We again create a simple a three-layer feed-forward neural network with ReLU activation [42], using 50 neurons per layer, and a learning rate of 0.001. The model takes as input our previously computed neural embed-

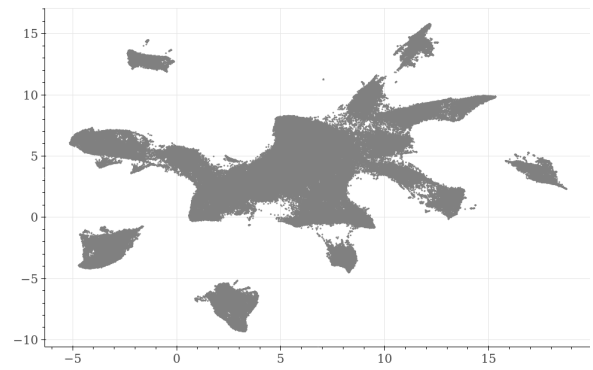


Figure 5: *Visualization of the unlabeled ASV21 data via 2D UMAP projection to cluster our neural attack signatures.*

dings as the attack signatures. We use our signatures as feature vectors while attempting to classify attack types. We train on 90% of the ASV19 data while evaluating on the other 10%. Our simple classifier achieves a test accuracy of 97.10% for the *in-domain* evaluation. The classification accuracy achieved by our neural embedding attack signatures (97.10%) significantly outperforms what we had achieved with our low-level signal feature attack signatures (56.96%).

We applied our neural embedding attack signature technique to a completely held-out dataset, from the ASVspoofer 2021 challenge (ASV21). It is similar to the ASV19 dataset but introduces more samples, new attackers, and a variety of codecs which blur the characteristics of spoofed audio that our model trained on. The attacker labels have not yet been released therefore we leave to future work an extensive evaluation once the labels have been released. We can apply our ASV19-trained model to the dataset and inspect the unlabeled clustering. This is shown in fig. 5. We can see that our model succeeds in finding various clusters, which indicates that our model generalizes beyond the ASV19 dataset.

## 5. Conclusion and Future Work

In this paper we introduced two new methods for creating attacker signatures to attribute spoofed audio to specific attackers. We evaluated signatures from low-level signal features and neural embeddings. We found that neural embeddings are well-suited for this problem and perform better in clustering and classification than low-level signal features. We have also demonstrated how our methodology can be applied to a completely held-out dataset and we suggest future work to investigate model adaptation once the labels have been released for the ASV21 dataset. Our model produces distinct clusters, indicating that it has learned to generalize beyond the ASV19 dataset. The neural embedding attack signatures are a very promising avenue for future attacker attribution research.

## 6. References

- [1] R. Spivak, “‘Deepfakes’: The Newest Way to Commit One of the Oldest Crimes,” *Geo. L. Tech. Rev.*, vol. 3, p. 339, 2018.
- [2] J. Botha and H. Pieterse, “Fake News and Deepfakes: A Dangerous Threat for 21st Century Information Security,” in *ICCWS 2020 15th International Conference on Cyber Warfare and Security*, 2020, p. 57.
- [3] “A Voice Deepfake Was Used To Scam A CEO Out Of \$243,000,” <https://www.forbes.com/sites/jessedamiani/2019/09/03/a-voice->

- deepfake-was-used-to-scam-a-ceo-out-of-243000/, (Accessed on 01/05/2022).
- [4] "ASVspoof: Automatic Speaker Verification Spoofing And Countermeasures Challenge," <https://www.asvspoof.org>, accessed: 2021-06-11.
  - [5] T. Rid and B. Buchanan, "Attributing Cyber Attacks," *Journal of Strategic Studies*, vol. 38, no. 1-2, pp. 4–37, 2015.
  - [6] F. J. Egloff and A. Wenger, "Public Attribution of Cyber Incidents," *CSS Analyses in Security Policy*, vol. 244, 2019.
  - [7] L. Maglaras, M. A. Ferrag, A. Derhab, M. Mukherjee, H. Janicke, and S. Rallis, "Threats, Protection and Attribution of Cyber Attacks on Critical Infrastructures," *arXiv preprint arXiv:1901.03899*, 2019.
  - [8] F. Skopik and T. Pahi, "Under False Flag: Using Technical Artifacts for Cyber Attack Attribution," *Cybersecurity*, vol. 3, no. 1, pp. 1–20, 2020.
  - [9] F. J. Egloff and M. Smeets, "Publicly Attributing Cyber Attacks: A Framework," *Journal of Strategic Studies*, pp. 1–32, 2021.
  - [10] S. Code, "We Are Truly Fucked: Everyone Is Making AI-Generated Fake Porn Now," <https://www.vice.com/en/article/bjye8a/reddit-fake-porn-app-daisy-ridley>, (Accessed on 01/05/2022).
  - [11] D. Wen, H. Han, and A. K. Jain, "Face Spoof Detection with Image Distortion Analysis," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 4, pp. 746–761, 2015.
  - [12] I. Amerini, L. Galteri, R. Caldelli, and A. Del Bimbo, "Deepfake Video Detection Through Optical Flow Based CNN," in *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, 2019, pp. 0–0.
  - [13] B. Dolhansky, R. Howes, B. Pflaum, N. Baram, and C. C. Ferrer, "The Deepfake Detection Challenge (DFDC) Preview Dataset," *arXiv preprint arXiv:1910.08854*, 2019.
  - [14] Y. Wang, R. Skerry-Ryan, D. Stanton, Y. Wu, R. J. Weiss, N. Jaitly, Z. Yang, Y. Xiao, Z. Chen, S. Bengio, Q. Le, Y. Agiomirgiannakis, R. Clark, and R. A. Saurous, "Tacotron: Towards End-to-End Speech Synthesis," in *Proceedings of Interspeech 2017*, 2017, pp. 4006–4010.
  - [15] D. Griffin and J. Lim, "Signal Estimation from Modified Short-Time Fourier Transform," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 32, no. 2, pp. 236–243, 1984.
  - [16] A. van den 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," in *Proc. 9th ISCA Workshop on Speech Synthesis Workshop (SSW 9)*, 2016, p. 125.
  - [17] J. Shen, R. Pang, R. J. Weiss, M. Schuster, N. Jaitly, Z. Yang, Y. Xiao, Y. Zhang, Y. Wang, R. Skerry-Ryan et al., "Natural TTS Synthesis by Conditioning WaveNet on Mel Spectrogram Predictions," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 4779–4783.
  - [18] S. Choi, S. Han, D. Kim, and S. Ha, "Attention: Few-Shot Text-to-Speech Utilizing Attention-Based Variable-Length Embedding," in *Proceedings of Interspeech 2020*, 2020.
  - [19] Y. Ren, Y. Ruan, X. Tan, T. Qin, S. Zhao, Z. Zhao, and T.-Y. Liu, "Fastspeech: Fast, Robust and Controllable Text to Speech," *Advances in Neural Information Processing Systems*, vol. 32, 2019.
  - [20] Y. Wang, D. Stanton, Y. Zhang, R.-S. Ryan, E. Battenberg, J. Shor, Y. Xiao, Y. Jia, F. Ren, and R. A. Saurous, "Style Tokens: Unsupervised Style Modeling, Control and Transfer in End-to-End Speech Synthesis," in *International Conference on Machine Learning*, 2018, pp. 5180–5189.
  - [21] A. Chintha, B. Thai, S. J. Sohrawardi, K. Bhatt, A. Hickerson, M. Wright, and R. Ptucha, "Recurrent Convolutional Structures for Audio Spoof and Video Deepfake Detection," *IEEE Journal of Selected Topics in Signal Processing*, vol. 14, 2020.
  - [22] M. Alzantot, Z. Wang, and M. B. Srivastava, "Deep Residual Neural Networks for Audio Spoofing Detection," *Proceedings of Interspeech 2019*, pp. 1078–1082, 2019.
  - [23] B. Chettri, D. Stoller, V. Morfi, M. A. M. Ramirez, E. Benetos, and B. L. Sturm, "Ensemble Models for Spoofing Detection in Automatic Speaker Verification," *Proceedings of Interspeech 2019*, pp. 1018–1022, 2019.
  - [24] Z. Wang, S. Cui, X. Kang, W. Sun, and Z. Li, "Densely Connected Convolutional Network for Audio Spoofing Detection," in *2020 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*. IEEE, 2020, pp. 1352–1360.
  - [25] N. Müller, F. Dieckmann, P. Czempin, R. Canals, K. Böttinger, and J. Williams, "Speech is Silver, Silence is Golden: What do ASVspoof-trained Models Really Learn?" in *Proceedings of 2021 Automatic Speaker Verification and Spoofing Countermeasures Challenge*, 2021, pp. 55–60.
  - [26] J. Williams and J. Rownicka, "Speech Replay Detection with x-Vector Attack Embeddings and Spectral Features," *Proceedings of Interspeech 2019*, pp. 1053–1057, 2019.
  - [27] B. Khoo, R. C.-W. Phan, and C.-H. Lim, "Deepfake Attribution: On the Source Identification of Artificially Generated Images," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, p. e1438, 2021.
  - [28] B. Zhang, J. P. Zhou, I. Shumailov, and N. Papernot, "On Attribution of Deepfakes," *arXiv preprint arXiv:2008.09194*, 2020.
  - [29] Y. Zhao, R. Togneri, and V. Sreeram, "Spoofing Detection Using Adaptive Weighting Framework and Clustering Analysis," in *Proceedings of Interspeech 2018*, 2018, pp. 626–630.
  - [30] Y. Jadoul, B. Thompson, and B. de Boer, "Introducing Parselmouth: A Python interface to Praat," *Journal of Phonetics*, vol. 71, pp. 1–15, 2018.
  - [31] "Praat: Doing Phonetics by Computer," <https://www.fon.hum.uva.nl/praat/>, (Accessed on 01/05/2022).
  - [32] J. Robert, M. Webbie et al., "Pydub," 2018. [Online]. Available: <http://pydub.com/>
  - [33] F. Eyben and B. Schuller, "openSMILE:) The Munich open-source large-scale multimedia feature extractor," *ACM SIGMultimedia Records*, vol. 6, no. 4, pp. 4–13, 2015.
  - [34] Y. Gao, J. Lian, B. Raj, and R. Singh, "Detection and Evaluation of Human and Machine Generated Speech in Spoofing Attacks on Automatic Speaker Verification Systems," in *2021 IEEE Spoken Language Technology Workshop (SLT)*, 2021, pp. 544–551.
  - [35] "UVA.NL Website: Voice 3. Shimmer," [www.fon.hum.uva.nl/praat/manual/Voice\\_3...Shimmer.html](http://www.fon.hum.uva.nl/praat/manual/Voice_3...Shimmer.html), (Accessed on 03/09/2022).
  - [36] "The LJ Speech Dataset," <https://keithito.com/LJ-Speech-Dataset/>, (Accessed on 01/05/2022).
  - [37] "pydub/api.markdown at master · jjaaro/pydub," <https://github.com/jjaaro/pydub/blob/master/API.markdown>, (Accessed on 03/09/2022).
  - [38] J. Sanchez, I. Saratxaga, I. Hernaez, E. Navas, D. Erro, and T. Raitio, "Toward a Universal Synthetic Speech Spoofing Detection Using Phase Information," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 4, pp. 810–820, 2015.
  - [39] Y. Jia, Y. Zhang, R. Weiss, Q. Wang, J. Shen, F. Ren, P. Nguyen, R. Pang, I. Lopez Moreno, Y. Wu et al., "Transfer Learning from Speaker Verification to Multispeaker Text-to-Speech Synthesis," *Advances in Neural Information Processing Systems*, vol. 31, 2018.
  - [40] J. S. Chung, J. Huh, S. Mun, M. Lee, H.-S. Heo, S. Choe, C. Ham, S. Jung, B.-J. Lee, and I. Han, "In Defence of Metric Learning for Speaker Recognition," in *Proceedings of Interspeech 2020*, 2020.
  - [41] L. McInnes, J. Healy, and J. Melville, "Umap: Uniform manifold approximation and projection for dimension reduction," *arXiv preprint arXiv:1802.03426*, 2018.
  - [42] A. F. Agarap, "Deep Learning Using Rectified Linear Units (ReLU)," *arXiv preprint arXiv:1803.08375*, 2018.